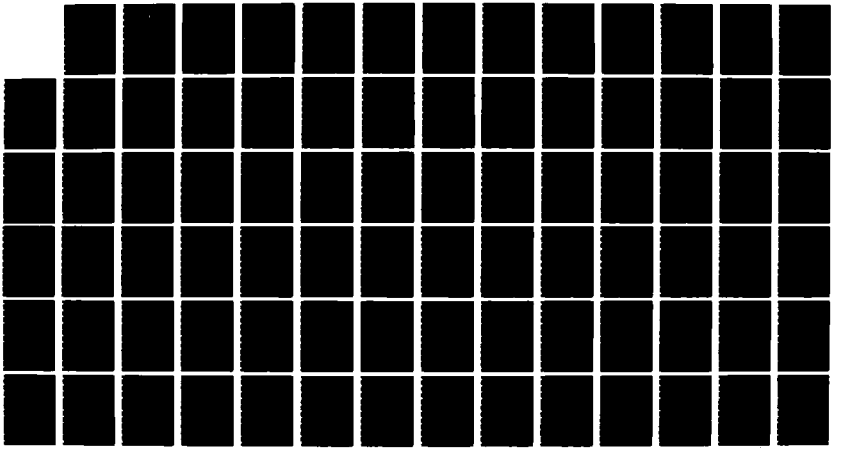
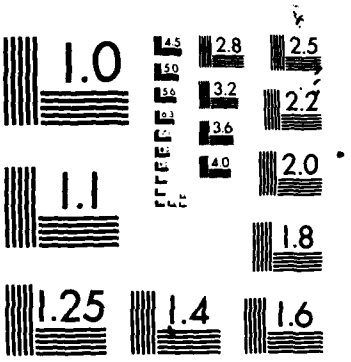
AD-A172 719 FEASIBILITY AND COMPARISON INVESTIGATION OF THE USE OF 1/1
THE FAST FOURIER T.. (U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. T JONES
UNCLASSIFIED MAR 86 AFIT/GNE/ENP/86M-8 F/G 12/1 NL





FEASIBILITY AND COMPARISON INVESTIGATION OF THE USE OF THE FAST FOURIER TRANSFORM AND FINITE DIFFERENCE METHOD FOR NUMERICAL SOLUTION OF BOUNDARY VALUE PROBLEMS

THESIS

Todd R. Jones
Major, USA

AFIT/GNE/ENP/86M-8

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GNE/ENP/86M-8

1

FEASIBILITY AND COMPARISON INVESTIGATION OF THE USE OF THE FAST FOURIER TRANSFORM AND FINITE DIFFERENCE METHOD FOR NUMERICAL SOLUTION OF BOUNDARY VALUE PROBLEMS

THESIS

Todd R. Jones
Major, USA

AFIT/GNE/ENP/86M-8

Approved for public release; distribution unlimited

AFIT/GNE/ENP/86M-8

FEASIBILITY AND COMPARISON INVESTIGATION OF THE USE OF THE FAST FOURIER TRANSFORM AND FINITE DIFFERENCE METHOD FOR NUMERICAL SOLUTION OF BOUNDARY VALUE PROBLEMS

THESIS

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Nuclear Engineering

Todd R. Jones, B.S.

Major, USA

March 1986

Preface

The purpose of this study was to determine the feasibility of using Fast Fourier Transforms (FFT) to solve Boundary Value Problems (BVP). Since many Boundary Value Problems are solved using some type of Finite Difference Method, it was felt that a comparison between these two methods might provide an insight into the usefulness of the Fast Fourier Transform in solving BVP.

The one dimensional BVP was studied to help provide a basis for understanding the two dimensional BVP. The two dimensional BVP was studied using only a simple case where the boundary conditions were zero, but could be easily extended to non-homogeneous boundary conditions.

The understanding and insight of the FFT I gained these last twelve weeks has been extraordinary. My thanks to Dr. N. Pagano of the Air Force Weapons Lab for sponsoring this thesis. I would like to acknowledge Dr. Kaplan for his never ending support, even when the prospects of getting the one and two dimensional FFT computer codes working was sometimes questionable. I would also like to thank Cpt. Ric Routh and Cpt. Jim Helton who provided greater understanding and appreciation for the FFTs. And finally, I must thank my wife, Jeanine, and my children, Jennifer, Greg, and Derek, who supported and sustained me through this period.

Todd R. Jones

Table of Contents

List of Figures

List of Tables

Abstract

The purpose of this study was to determine the feasibility of using Fast Fourier Transforms (FFT) to solve Boundary Value Problems (BVP) and then compare the results to those of the Finite Difference Method (FDM). Variations of Poisson's one and two dimensional equations were used as a vehicle to develop the FFT method. For the one dimensional BVP, both homogeneous and non-homogeneous Dirichlet boundary conditions were considered. In the one dimensional BVP the inhomogeneous function, F(x), was also varied. The two dimensional BVP, only one inhomogeneous function, F(x,y), and homogeneous boundary conditions were used. The one dimensional model was used as a basis for developing the two dimensional model.

The analytical solution of each problem was compared to the numerical solution of the FDM and the FFT method at varying mesh sizes. The computational time of the FDM and the FFT method were also compared.

The results indicate that the FFT is extremely efficient in the two dimensional BVP because of the computer storage space required and the computational time needed to solve the FFTs. The accuracy of the FFT compares favorably to the FDM and, as the mesh size decreases, becomes more accurate than the FDM.

FEASIBILITY AND COMPARISON INVESTIGATION OF THE USE OF THE FAST FOURIER TRANSFORM AND FINITE DIFFERENCE METHOD FOR NUMERICAL SOLUTION OF BOUNDARY VALUE PROBLEMS

I. Introduction

Background

Many science and engineering problems require the solution of one or more Boundary Value Problems (BVP). Many of these BVPs cannot be solved analytically because of irregular boundary conditions or complex geometries. The most common method of solving these type of BVPs is by means of the Finite Difference Method (FDM). In this method the derivatives of the partial differential equation are approximated by use of Taylor series expansion, which reduces it to a set of algebraic equations that can be solved by simultaneous equations. The solution of these simultaneous equations may be found with the aid of a computer by using direct matrix inversion techniques which require N^3 operations, where N is the number of simultaneous equations (14:6). Hence, other methods which require fewer operations have been developed to solve sets of simultaneous linear equations. These methods include both direct and iterative techniques and in each case decrease the computational time (3:111; 6:417).

One of the techniques used to solve BVPs involve using Discrete Fourier Transforms (DFT). The computational time for a DFT is only on the order N^2, but to solve a one dimensional BVP it becomes N^3. When the

Fast Fourier Transform (FFT) algorithm is applied to the DFT the computational time is decreased to an order of $NLOG_2N$ operations, which equates to $4NLOG_2N + 4N$ to solve the same one dimensional BVP (1:8; 11:215). This decrease in computational time greatly enhances the use of FFTs over direct FDMs such as Gauss Elimination. The use of the DFT in conjunction with the FFT will be referred to as the FFT method throughout the remainder of this study, since the DFT takes advantage of the FFT algorithm for increasing its computational speed.

Purpose

This thesis topic stems from an article in the May 1984 Physics Today (5), which mentioned the use of FFTs to solve BVPs in two and three dimensions. For a three dimensional BVP with a grid or mesh of 100x100x100, the matrix becomes very large, namely $10^6 \times 10^6$. This size matrix, according to the article, can easily be solved using FFT techniques (5:56). A closer investigation of this method revealed a lack of published information concerning the FFT technique. It was found that the FFT method was faster and required less computer space than conventional FDMs for two and three dimensions (5:547; 13:710). In the information that was published only BVPs with homogeneous boundary conditions were solved. There is an obvious lack of information on the feasibility of using FFTs to solve BVPs in one, two, and three dimensions. Additionally, there is no known published information concerning the use of FFTs to solve BVPs with boundary conditions other than zero.

This study investigated the feasibility of using FFTs to solve BVPs and compared this method with the Gauss Elimination method in terms of both computational speed and accuracy.

Scope

The study will consider only the problem of the one and two dimensional Poisson's Equation, with Dirichlet Boundary Conditions. Both zero and nonzero boundary conditions will be analyzed. In each dimension the Poisson's equation will be solved analytically, and numerically using Gauss Elimination and FFTs. The accuracy of the Gauss Elimination method and the FFT method will be compared to the analytical solution. The computational time of the Gauss Elimination and the FFT method will then be compared.

Plan of Development

The initial approach was to develop a FFT computer program to solve a one dimensional Poisson Equation with boundary conditions equal to zero. A Gauss Elimination (GE) program was then developed to solve the same one dimensional Poisson Equation with boundary conditions equal to zero. These two programs were then expanded to accept a two dimensional Poisson equation with boundary conditions equal to zero. Modifications were then made to account for boundary conditions other than zero.

The accuracy of the numerical solutions were then compared to the analytical solutions, and computational times of the GE method were compared to the computational times of FFT method. Finally, limitations on the FFT method and feasibility of possible directions of further research were discussed.

II. Theory

In order to understand the use of FFTs to solve boundary value problems, it is necessary to understand some theory about the Fourier Series and the Discrete Fourier Transform.

Fourier Series

Any function expanded as a series of eigenfunctions is defined as a Fourier Series, where the interval of orthogonality is $(0 \leq x \leq 2\pi)$ (17:65). Because the series is periodic any 2π length can be used. If the interval $(0 \leq x \leq 2\pi)$ is replaced by $(-L \leq x \leq L)$, then the Fourier Series can be defined as

$$U(x) = a_o/2 + \sum_{n=1}^{\infty} [a_n \cos(n\pi x/L) + b_n \sin(n\pi x/L)] \qquad (2.1)$$

The coefficients a_o, a_n and b_n are

$$a_o = 1/L \int_{-L}^{L} U(x)dx \qquad (2.2)$$

$$a_n = 1/L \int_{-L}^{L} U(x)\cos(n\pi x/L)dx \quad n=1,2,3. \ . \ . \qquad (2.3)$$

$$b_n = 1/L \int_{-L}^{L} U(x)\sin(n\pi x/L)dx \quad n=1,2,3. \ . \ . \qquad (2.4)$$

This same series can be moved along any interval $(-L \leq x \leq L)$ or from $(0 \leq x \leq 2L)$ as long as the interval remains 2π (7:283). Thus the Fourier coefficients can be rewritten as

$$a_o = 2/L \int_o^L U(x)dx \tag{2.5}$$

$$a_n = 2/L \int_o^L U(x)\cos(n\pi x/L)dx \quad n=1,2,3. \ . \ . \tag{2.6}$$

$$b_n = 2/L \int_o^L U(x)\sin(n\pi x/L)dx \quad n=1,2,3. \ . \ . \tag{2.7}$$

Discrete Fourier Series

The Fourier Series can also be expressed as a discrete finite series. The derivation will not be shown here, but can be found in Numerical Analysis books by Richard W. Hamming (7), or by Robert Vichnevetsky (16). There are three orthogonality relationships, though, that aid in the derivation of the Discrete Fourier Series and understanding of the Discrete Fourier Transform (7:284).

$$\sum_{p=0}^{2N-1} \cos\{(2\pi/L)k(Lp/2N)\}\cos\{(2\pi/L)m(Lp/2N)\} = \begin{cases} 0 & k\neq m \\ N & k=m\neq 0 \\ 2N & k=m=0 \end{cases} \tag{2.8}$$

$$\sum_{p=0}^{2N-1} \cos\{(2\pi/L)k(Lp/2N)\}\sin\{(2\pi/L)m(Lp/2n)\} = 0 \tag{2.9}$$

$$\sum_{p=0}^{2N-1} \sin\{(2\pi/L)k(Lp/2N)\}\sin\{(2\pi/L)m(Lp/2N)\} = \begin{cases} 0 & k\neq m \\ N & k=m\neq 0 \\ 2N & k=m=0 \end{cases} \tag{2.10}$$

Based on these orthogonality relationships the Discrete Fourier Series can be defined as

$$U(x) = a_o/2 + \sum_{k=1}^{N-1} \{a_k \cos(k2\pi x/L) + b_k \sin(k2\pi x/L)\} \qquad (2.11)$$

where

$$a_k = 1/N \sum_{k=0}^{2N-1} U(x)\cos(2\pi kx/L) \qquad (2.12)$$

$$b_k = 1/n \sum_{k=0}^{2N-1} U(x)\sin(2\pi kx/L) \qquad (2.13)$$

The coefficients a_k and b_k are also called the Discrete Fourier Transforms (16:50). To see why this is true it is necessary to review the Fourier Integral, then relate the Integral to the Discrete Fourier Transform.

Discrete Fourier Transform

For the purposes of this study the Fourier Integral will not be derived, but only stated. Several books contain the derivation of this Integral, which include E.C. Titchmarch's book on Fourier Integrals (15). The Fourier Integral is defined as

$$U(w) = \int_{-\infty}^{\infty} U(t)e^{-iwt}\, dt \qquad (2.14\text{-a})$$

then

$$U(t) = 1/2\pi \int_{-\infty}^{\infty} U(w)e^{iwt}\, dw \qquad (2.14\text{-b})$$

where $w = 2\pi f$. The function U(w) is called the Fourier Transform of U(t), and U(t) is the inverse Fourier Transform of U(w). Equation (2.14-a)

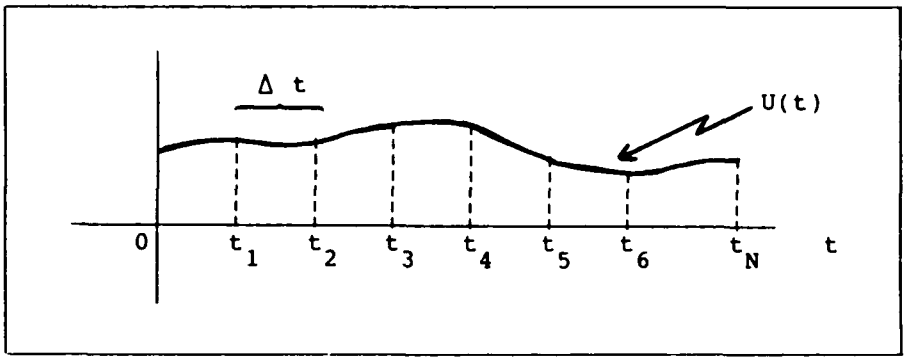


Figure 1. Function U(t) on the Interval ($0 \leq t \leq t_N$)

and (2.14-b), also known as the Fourier Integral Theorem, can then be discretized to obtain the Discrete Fourier Transform.

Let U(t) be defined as some function between the interval ($0 \leq t \leq t_N$) as outlined in Figure 1. One can then determine the Discrete Fourier Transform of U(t), which is defined as U(w).

If one allows U(t) to be discrete for every $U(t_k)$, where $k=0,1,2 \ldots N$, and applies the Fourier Integral Theorem, while letting "FT" be defined as a "Fourier Transform," the following is obtained.

$$FT\{U(t)\} = U(w) = \int_{-\infty}^{\infty} U(t)e^{-iwt}\,dt \qquad (2.15)$$

In equation (2.15) w is defined as the frequency $w=2\pi/T$, and T is defined as the length from 0 to t_N. Because $U(t)=0$ for $t<0$ then

$$U(w) = \int_{0}^{\infty} U(t)e^{-iwt}\,dt \qquad (2.16)$$

By letting $U(t)=0$ for $t>t_N$, then

$$U(w) = \int_{o}^{t_N} U(t)e^{-iwt}\,dt \qquad (2.17)$$

By discretizing the integral the following is obtained

$$U(w) = \int_{o}^{t_N} U(t)e^{-iwt}\,dt \qquad (2.18\text{-a})$$

$$U(w) \approx \sum_{k=0}^{N-1} U(t_k)e^{-iwt_k}\,\Delta t \qquad (2.18\text{-b})$$

where $t_N=N\Delta t=T$, since N is defined as the number of intervals between zero and t_N and Δt is defined as the interval. This means $\Delta t=T/N$ and $t_k=k\Delta t=kT/N$, then

$$U(w) = \sum_{k=0}^{N-1} U(t_k)e^{-iw(kT/N)}\,T/N \qquad (2.19\text{-a})$$

$$U(w) = T/N \sum_{k=1}^{N} U(t_k)e^{-iw(kT/N)} \qquad (2.19\text{-b})$$

Since $w=2\pi/T$, then $w=m\Delta w=m2\pi/T$, where $m=1,2,3.\ .\ .\ .N$, which leads to

$$U(m\Delta w) = T/N \sum_{k=0}^{N-1} U(t_k)e^{-i(2\pi m/T)\,(kT/N)} \qquad (2.20\text{-a})$$

$$U(m\Delta w) = T/N \sum_{k=0}^{N-1} U(t_k)e^{-i2\pi mk/N} \qquad (2.20\text{-b})$$

$$U(m\Delta w) = U(w_m) = (T)FT\{U(t_k)\} \qquad (2.20\text{-}c)$$

where the Discrete Fourier Transform is defined as

$$U(w_m) = FT\{U(t_k)\} = 1/N \sum_{k=0}^{N-1} U(t_k)e^{-i2\pi km/N} \qquad (2.21\text{-}a)$$

If one lets "FT^{-1}" be defined as the Inverse Discrete Fourier Transform, then

$$U(t_k) = FT^{-1}\{U(w_m)\} = \sum_{m=0}^{N-1} U(w_m)e^{i2\pi km/N} \qquad (2.21\text{-}b)$$

The Discrete Fourier Transform in equation (2.21-a) is called the complex form of the DFT. The DFT can also be expressed in terms of sines and cosines by using Euler's identity, $e^{ijx} = \cos(jx) + i\sin(jx)$, and is defined as

$$U(w_m) = a_o/2 + \sum_{k=1}^{N-1} [a_k\cos(2\pi km/N) + b_k\sin(2\pi km/N)] \qquad (2.22)$$

It is obvious, then, from equation (2.11) that a_o, a_k and b_k are the same Fourier coefficients. Thus, these Fourier coefficients are also called the Discrete Fourier Transforms (16:50).

Sampling and Aliasing

There are two terms that are synonymous with DFTs that need an explanation. Since the DFT is not continuous, a discrete finite number of points must be determined at which the DFT will be calculated. These

equidistant points are called the "sample" over which the DFT is evaluated. For example, U(t), in Figure 1, is being "sampled" over the interval $t_k=t_1, t_2, t_3 . . . . . t_N$.

The other term that needs explanation is aliasing. If Δt, in Figure 1, is too large, the Fourier coefficients of the higher frequencies will fold into the coefficients of the lower frequencies. For example, if a sample of eight points is taken over an interval, the coefficient of the first harmonic will be equal to the coefficient of the seventh harmonic; the coefficient of the second harmonic will be equal to the coefficient of the sixth harmonic; and the coefficient of the third and fifth harmonic will be equal. To avoid this problem of aliasing the Nyquist sampling rate is used. This formula is $1/\Delta t=2f$. Simply stated, this says if $\Delta t < 1/2f$, then aliasing will occur; if $t > 1/2f$ then aliasing will not occur (1:85). (The f is defined as the highest frequency component of the Fourier transform.)

Fast Fourier Transform

The Fast Fourier Transform (FFT) algorithm takes advantage of the symmetry of the trigonometric functions in the Discrete Fourier Transform (DFT). The regrouping of the equations in calculating the DFT reduces the number of computational operations. The Discrete Fourier Transform requires on the order of N^3 operations to solve a one dimensional BVP, where N is defined as the number of discrete data points. The Fast Fourier Transform algorithm is on the order of $4NLOG_2N + 4N$ operations to solve the same BVP. The FFT requires on the order of $NLOG_2N$ operations, but in calculating a one or two dimensional BVP the number of computations

must include calculations of the FFT, the calculations of the inverse FFT, and any computations conducted while the function is transformed. Additionally, any modifications to the FFT algorithm (i.e., deletions of the real or imaginary components of a complex array) add additional calculations (11:215). The theory behind the FFT algorithm will not be discussed in this study. E. Oran Brigham's book on FFTs (1) contains both an intuitive and theoretical development of the algorithm and is recommended to the reader who wishes to gain an indepth understanding on how and why the FFT algorithm works.

III. Poisson's Equation in One-Dimension

The first problem examined in this study is Poisson's equation in one dimension. The general form of the equation is

$$\frac{d^2U(x)}{dx^2} = -F(x) \qquad (3.1)$$

where U(x) is the unknown function to be determined and F(x) is a known function. Both homogeneous and non homogeneous boundary conditions are examined along with different values of F(x). The boundary conditions and the function F(x) are separated into four distinct cases.

Case 1	U(0)=0	U(L)=0	F(x) = 40
Case 2	U(0)=0	U(L)=0	F(x) = x
Case 3	U(0)=2	U(L)=8	F(x) = 40
Case 4	U(0)=2	U(L)=8	F(x) = x

Analytical Solution

The general solution to equation (3.1) for F(x) = 40 is found by direct integration and takes the form

$$U(x) = -20x^2 + Ax + B \qquad (3.2)$$

for case one and case three. Applying the boundary conditions in case one, equation (3.2) becomes

$$U(x) = -20x^2 + 200x \qquad (3.3)$$

and applying the boundary conditions in case three, equation (3.2) becomes

$$U(x) = -20x^2 + 200.6x + 2 \qquad (3.4)$$

The general solution to equation (3.1) for $F(x) = x$ in case two and four is also found by direct integration and takes the form

$$U(x) = \frac{-x^3}{6} + Cx + D \qquad (3.5)$$

By applying the boundary conditions in case two, equation (3.5) becomes

$$U(x) = \frac{-x^3}{6} + \frac{100x}{6} \qquad (3.6)$$

and applying the boundary conditions in case four, the equation becomes

$$U(x) = \frac{-x^3}{6} + \frac{259x}{15} + 2 \qquad (3.7)$$

Equations (3.3), (3.4), (3.6), and (3.7) are the analytical solutions to the one dimensional Poisson equation (3.1).

Numerical Approximation

This same equation (3.1) can be solved using numerical approximations. Two different techniques were examined, the FDM and the FFT method. Both methods involve the subdividing of the region concerned into N nodes, or mesh points, and solving a set of simultaneous equations for each value of N. The FDM can solve the simultaneous equations using different techniques including Gauss Elimination, Tridiagonal, or Iterative Methods. The FFT method uses the FFT algorithm, which solves the trigonometric equations using symmetry of the sine and cosine terms.

In general, the accuracy of the solution is dependent upon the number of nodal points chosen: the finer the mesh, the larger the number of nodes, and the greater the accuracy.

$$\begin{bmatrix} -2 & 1 & 0 & 0 \\ 1 & -2 & 1 & 0 \\ 0 & 1 & -2 & 1 \\ 0 & 0 & 1 & -2 \end{bmatrix} \begin{bmatrix} u(x_1) \\ u(x_2) \\ u(x_3) \\ u(x_4) \end{bmatrix} = -4 \begin{bmatrix} F(x_1) \\ F(x_2) \\ F(x_3) \\ F(x_4) \end{bmatrix}$$

Figure 2. General Solution to Equation (3.8) with 4 Interior Nodes

Finite Difference Method. The FDM, as mentioned in the Introduction, approximates the derivatives of the differential equation by use of a truncated Taylor Series expansion. The derivation can be found in many texts which include C.F. Gerald and P.O. Wheatley (6), and Clark and Hansen (3). By using the Taylor Series expansion a central difference approximation can be applied to equation (3.1) to get

$$\frac{d^2U(x)}{dx^2} \approx \frac{\delta^2U(x)}{h_x^2} = \frac{(U_{k+1} - 2U_k + U_{k-1})}{h_x^2} = -F(kh_x) \qquad (3.8)$$

where $x=kh_x$. The h_x is defined as the distance between nodal points, N is defined as the total number of interior nodes, and $k=1,2,3. \ . \ .N$ (14:6). Equation (3.8) can be reduced to matrix notation

$$Au = -F \qquad (3.9)$$

where "A" is defined as the coefficient matrix which is tridiagonal, "u" is a column matrix of unknowns, and "F" is a known column vector (see Figure 2). This matrix equation can then be solved by either the Tri-diagonal method (Thomas method) or the Gauss Elimination Method.

Fast Fourier Transform Method. The FDM takes advantage of a polynomial interpolation, whereas, the FFT method takes advantage of a trigonometric interpolation (16:89). In the FFT method, $h=L/(N+1)$, where L is defined as the length of the region of concern, N is the number of sample points and h is defined as the length of each sampled interval. It follows, that $x_n=nh$, where $n=0,1,2,3. \ . \ .N+1$ and is defined as the set of grid points in which the solution of equation (3.1) is to be approximated. By inspection the BVP must be an odd function, since at both boundary conditions the value is zero or a constant. The equation

$$U(x) = \sum_{k=1}^{N} b_k \sin(k\pi x/L) \quad k = 1,2,3. \ . \ .N \tag{3.10}$$

can be used to approximate the solution to equation (3.1). It follows from equation (2.11) and (2.13), and using the orthogonality relationship of equation (2.10) that

$$b_k = 2/(N+1) \sum_{k=1}^{N-1} U_k \sin(k\pi x/L) \tag{3.11}$$

By taking the second derivative of U(x), equation (3.10), with respect to x the following is found.

$$\frac{d^2U(x)}{dx^2} = - \sum_{k=1}^{N} b_k (k\pi/L)^2 \sin(k\pi x/L) \tag{3.12}$$

By substituting the right side of equation (3.12) into equation (3.1),

one gets the following results.

$$-\sum_{k=1}^{N} b_k(k\pi/L)^2\sin(k\pi x/L) = -F(x) \tag{3.13}$$

By discretizing x to x_n, where n=1,2,3. . .N then equation (3.13) becomes

$$\sum_{k=1}^{N} b_k(k\pi/L)^2\sin(k\pi x_n/L) = F(x_n) = -f_n \tag{3.14}$$

Now, one can solve for b_k using the orthogonality relationship, equation (2.10), to obtain

$$b_k = 2/(N+1)(L/k\pi)^2 \sum_{n=1}^{N} f_n\sin(k\pi x_n/L) \tag{3.15}$$

where

$$FT(f_n) = f_k = 2/(N+1) \sum_{n=1}^{N} f_n\sin(k\pi x_n/L) \tag{3.16}$$

Solve for U_n by discretizing U(x) such that

$$U_n = \sum_{k=1}^{N} b_k\sin(k\pi x_n/L) \tag{3.17}$$

The specific sequence used to solve for each nodal value of U_n using the FFT method is thus:

1. Compute the $FT(f_n)$, equation (3.16), by using the FFT algorithm.
2. Compute each value of b_k by dividing the eigenvalue of U(x) by the $FT(f_n)$. See equation (3.15).

3. Compute the nodal value of U_n by using the inverse FFT algorithm, equation (3.17).

Computer Analysis

FORTRAN codes were developed to solve the one dimensional Poisson's equation using the FDM and the FFT method. The codes take advantage of subroutines found in the International Mathematical and Statistical Libraries, Inc. library, more commonly called the IMSL library, and were run on the Harris 800 main frame computer. A listing of the codes using the FFT method are in Appendix A. The FDM used both the Thomas method and the Gauss Elimination method. The Thomas method was programed in FORTRAN using the algorithm found in Clark and Hensen, page 47. The Gauss Elimination method was programed in FORTRAN using the IMSL routine LEQIF.

FFT Algorithm. The IMSL subroutine used throughout this study of the FFT was FFTCC. The FFT algorithm requires a complex value input and provides a complex value output. Additionally it computes the transforms over a 2L interval. Because of these peculiarities it was necessary to modify the computer code to compute the FFT for the one dimensional Poisson's equation. Equation (3.1) is described only over an interval of L. In order to have the FFTCC subroutine operate properly over this interval it was necessary to input the function over a 2L interval, then only accept one half of the interval in the output as the solution. For example, in case one the function, $F(x)=40$, is a rectangular function over the interval $0 \leq x \leq L$. Since equation (3.1) is described by a sine function, which is odd, it was necessary to input $F(x)=40$ from $0 \leq x \leq L$

and $F(x)=-40$ from $L< x \leq 2L$. On output, only the REAL value of the complex ouput was accepted from $0\leq x \leq L$. In the FFTCC subroutine the first coefficient is only a_o. In electrical engineering terms this is referred to as the DC component. This component must be eliminated when solving the one dimensional Poisson's equation since the solution is only a series of sine terms. Hence, in the main FFT program it was necessary to sum for b_k from $n=2$ to N rather than $n=1$ to N.

$$b_k = \sum_{n=2}^{N} [2/(N+1)]\ [L/(k\pi)]^2\ f_n \sin(k\pi x_n/L) \qquad (3.18)$$

Equation (3.18) eliminates the DC component and allows the FFTCC subroutine to approximate the one dimensional Poisson's equation.

The four cases outlined on page 12 vary not only the function F(x), but also the boundary conditions. Two of the four cases used boundary conditions other than zero. In applying non-zero boundary conditions to the FDM the non-zero boundary is absorbed into the F(x) function. This process applies similarly to the FFT method. On input the function F(x) must be adjusted at the endpoint to account for the non-zero boundary condition. For example, in case three, over an interval from $0\leq x \leq L$, the values of F(x) would be as listed in Table I.

By programing the input values of F(x) in this manner the desired results are achieved. This process seems logical, as one examines a delta function, such as $F(0)=2$, in normal space, the value becomes a constant in Fourier space and bounds the equation that is transformed. When the inverse transform is conducted this constant returns to a delta function adjusted somewhat by the computation conducted in step two of the FFT method.

TABLE I

Values of F(x) vs x in Case 3
for the One-Dimensional BVP

Value of x	Value of F(x)
0	2
2	40
4	40
6	40
8	40
10	8

Exact Solution. Particular solutions to equation (3.1) were found by letting L=10 and solving for x at values of 2, 4, 6, and 8. The table of these results can be found in Appendix B. These results were then compared to the numerical solution using the FDM and the FFT method at the same values of x, 2, 4, 6, and 8.

Nodal Points. Each case of boundary conditions, as described on page 12, was computed using four different mesh sizes. The mesh size was decreased in each of the four cases, thus resulting in a better approximation to the analytical solution. The interior nodal mesh points used were 4, 9, 49, and 99. Appendix B contains the results of these computations.

Average Error. The accuracy of the approximations was determined using an average percent error, or relative error (6:42). For the one dimensional case the average percent error was defined as

$$\langle E \rangle = \left| \frac{\text{Exact Solution} - \text{Numerical Solution}}{\text{Exact Solution}} \right| \times 100 \qquad (3.19)$$

Comparison of Approximations to Exact Solutions. The plots of the average percent error for each set of interior nodes are found in Figures 3-6. In each of the four cases, as outlined on page 12, the FDM solution was found to be the same as the analytical solution. Consequently, the only error differences reflected on Figures 3-6 are in the FFT method. In all cases, as the number of interior nodes increase the average percent error in the FFT method decreases. The computations for each nodal point for the FDMs and the FFT method are in Appendix B, Tables III-VI.

Computational Time. In order to calculate N interior nodal points using the FFT method it is necessary to calculate the Fourier Transform of 2N+2 points. This is due to the interval having to be sampled over the entire 2L period as described in the paragraph on the FFT algorithm. A comparison of computing times is found in Figure 7. It is important to keep in mind that the FFT times reflect 2N+2 calculations while the FDMs are only N calculations. The Thomas method, or tridiagonal method, proves to be the most efficient method in the one dimensional problem. The FFT method becomes more efficient as the number of nodal points are increased as compared to the Gauss Elimination (GE) method. This is due to the $4NLOG_2N + 4N$ calculations required for the FFT method in one dimension, while the GE method requires N^3 calculations. Figure 8 is a comparison of the total computer time, which includes loading of arrays for input and writing the solutions to a file on output. The computation times for the algorithm and the computation times for the total computer program are listed in Appendix B, Tables VII and VIII respectively.

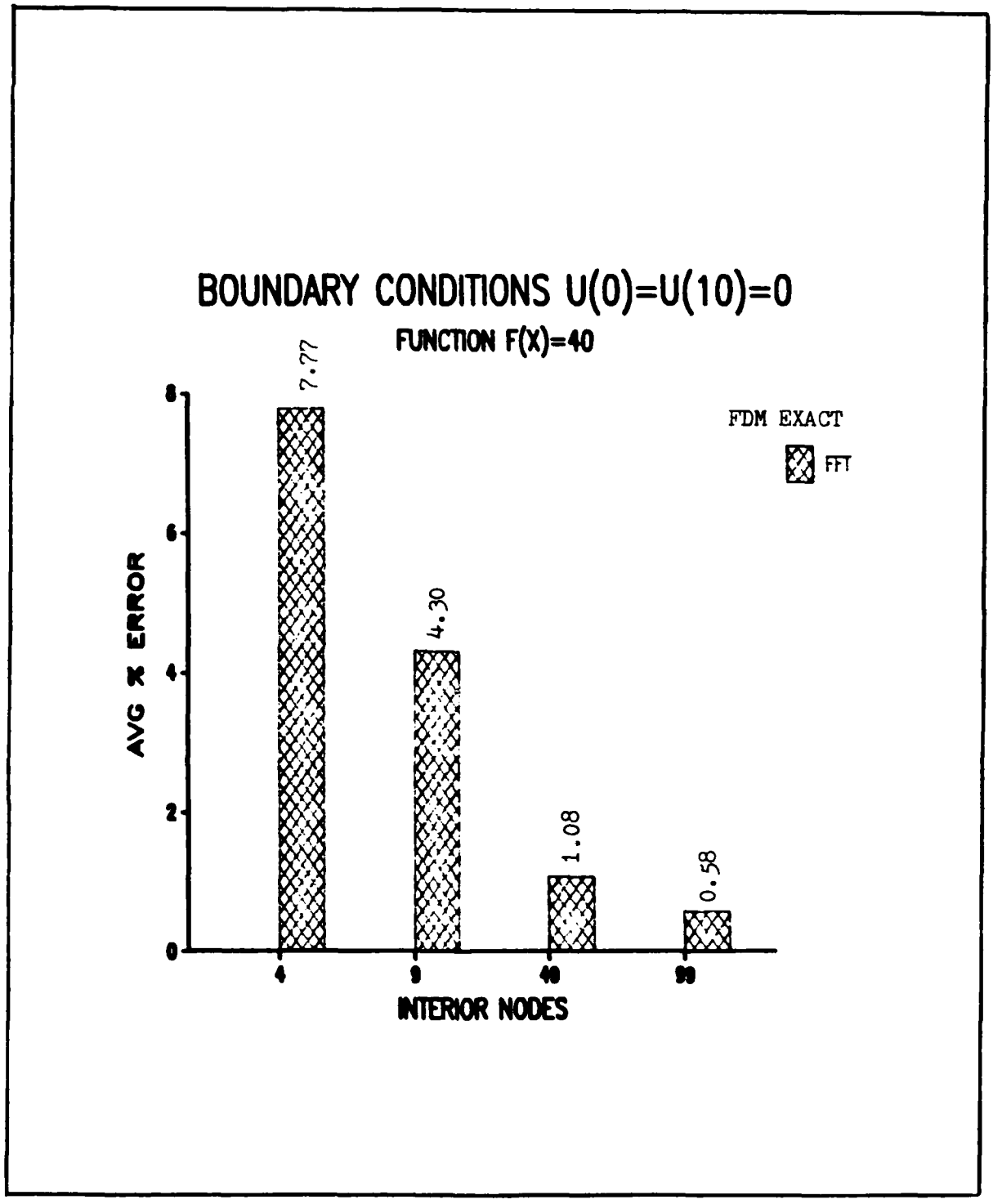


Figure 3. Case One, Average Error Comparison Between FDM, FFT Method and the Analytical Solution

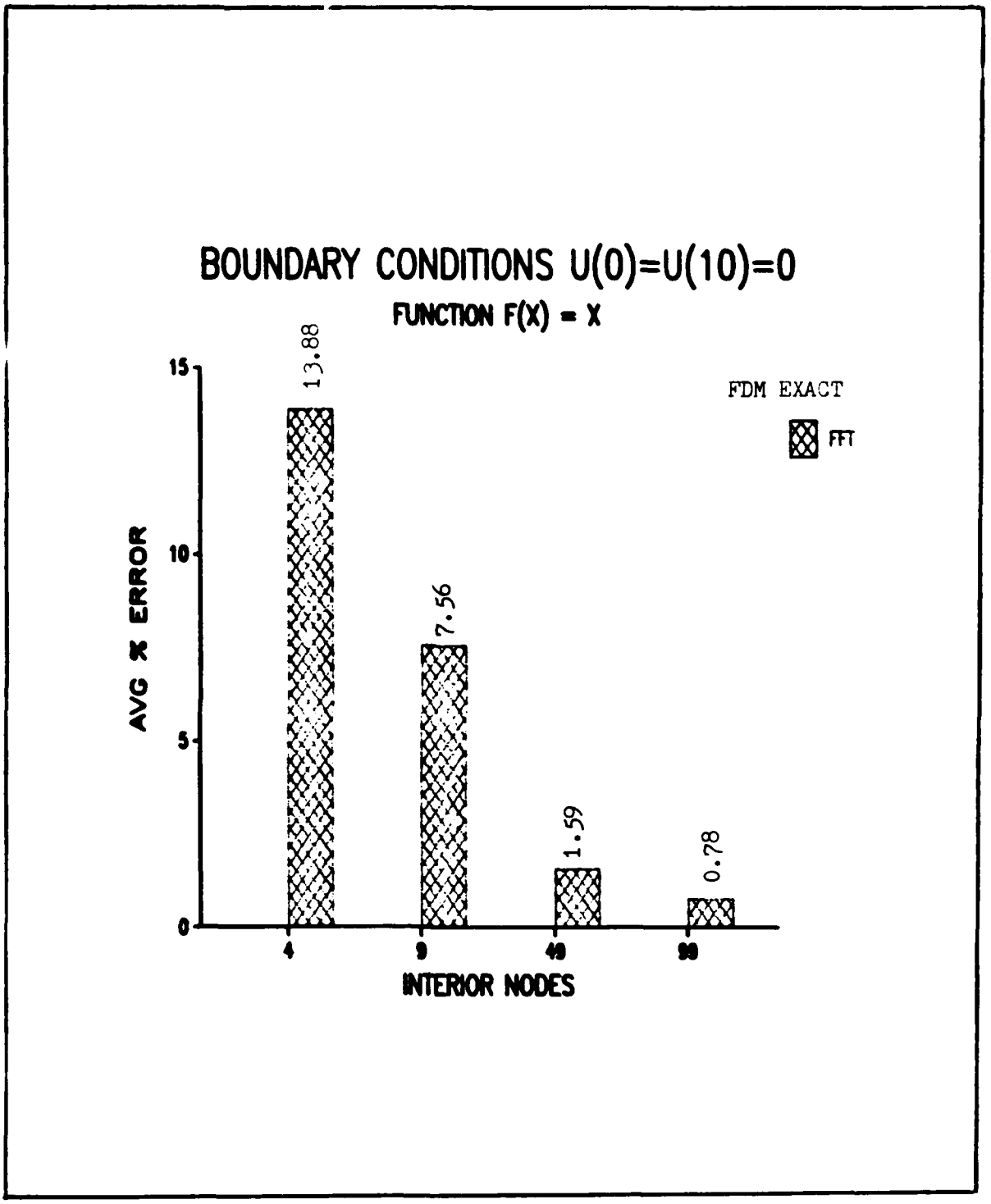


Figure 4. Case Two, Average Error Comparison Between FDM, FFT Method and the Analytical Solution

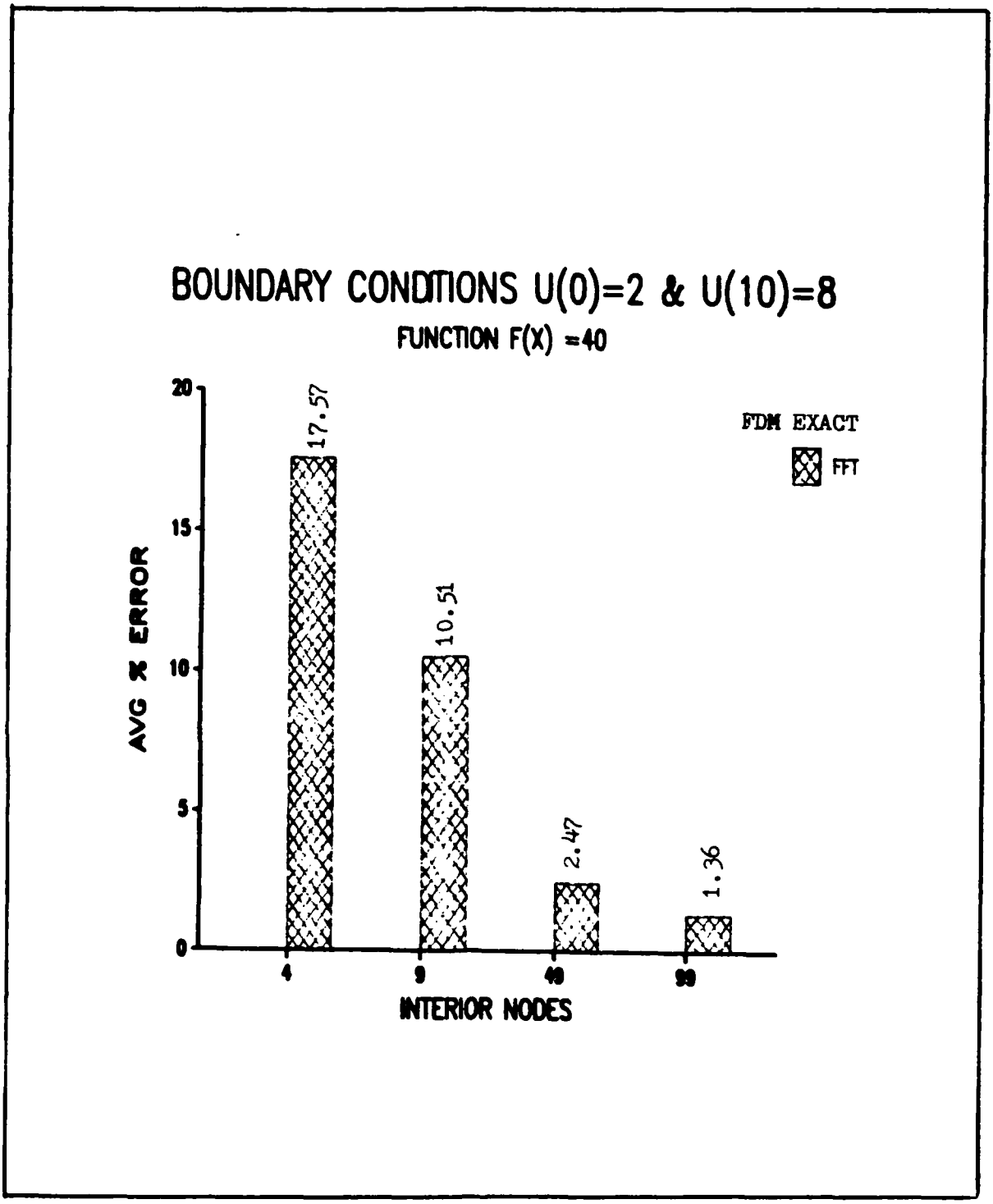


Figure 5. Case Three, Average Error Comparison Between FDM, FFT Method and the Analytical Solution

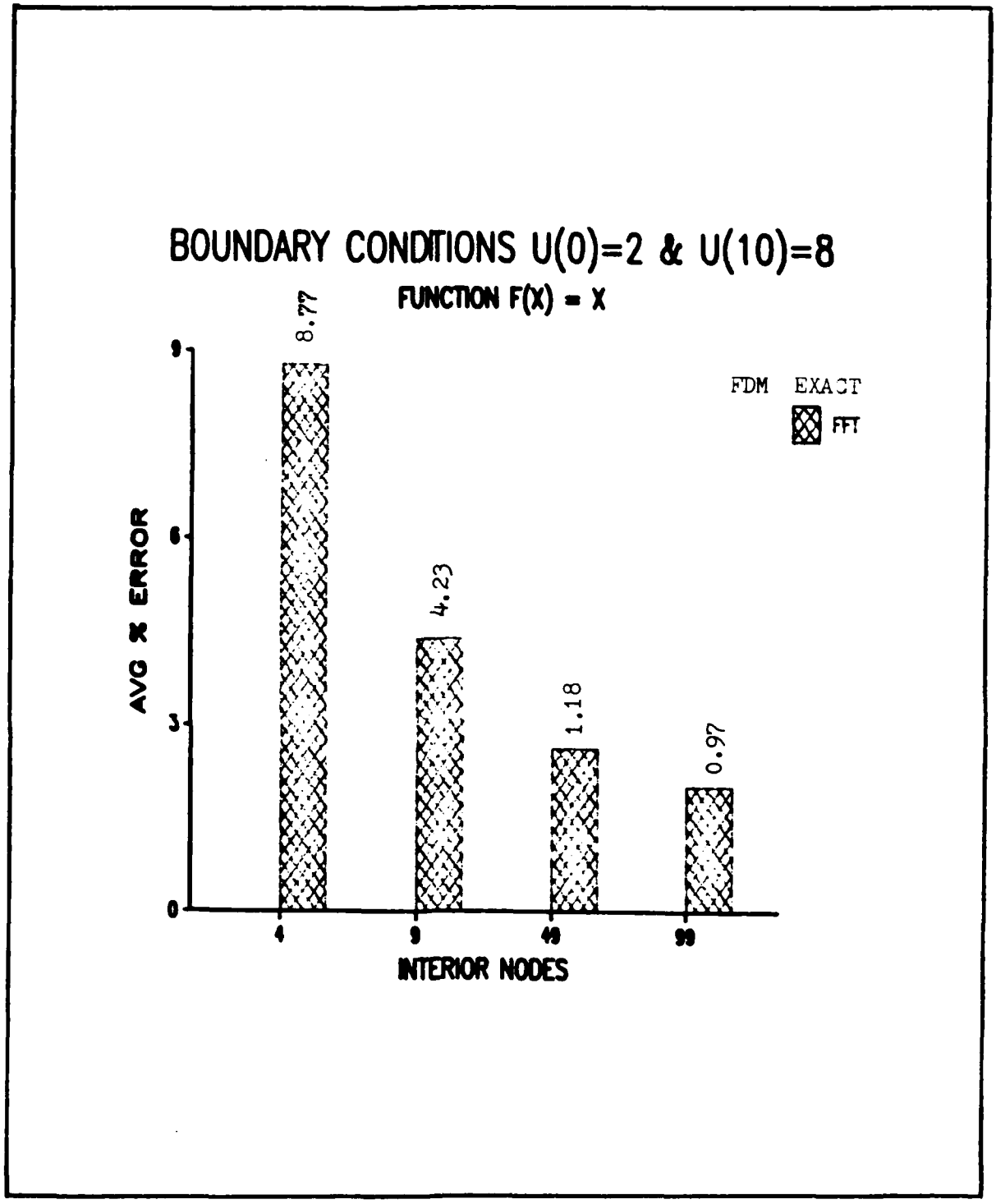


Figure 6. Case Four, Average Error Comparison Between FDM, FFT Method and the Analytical Solution

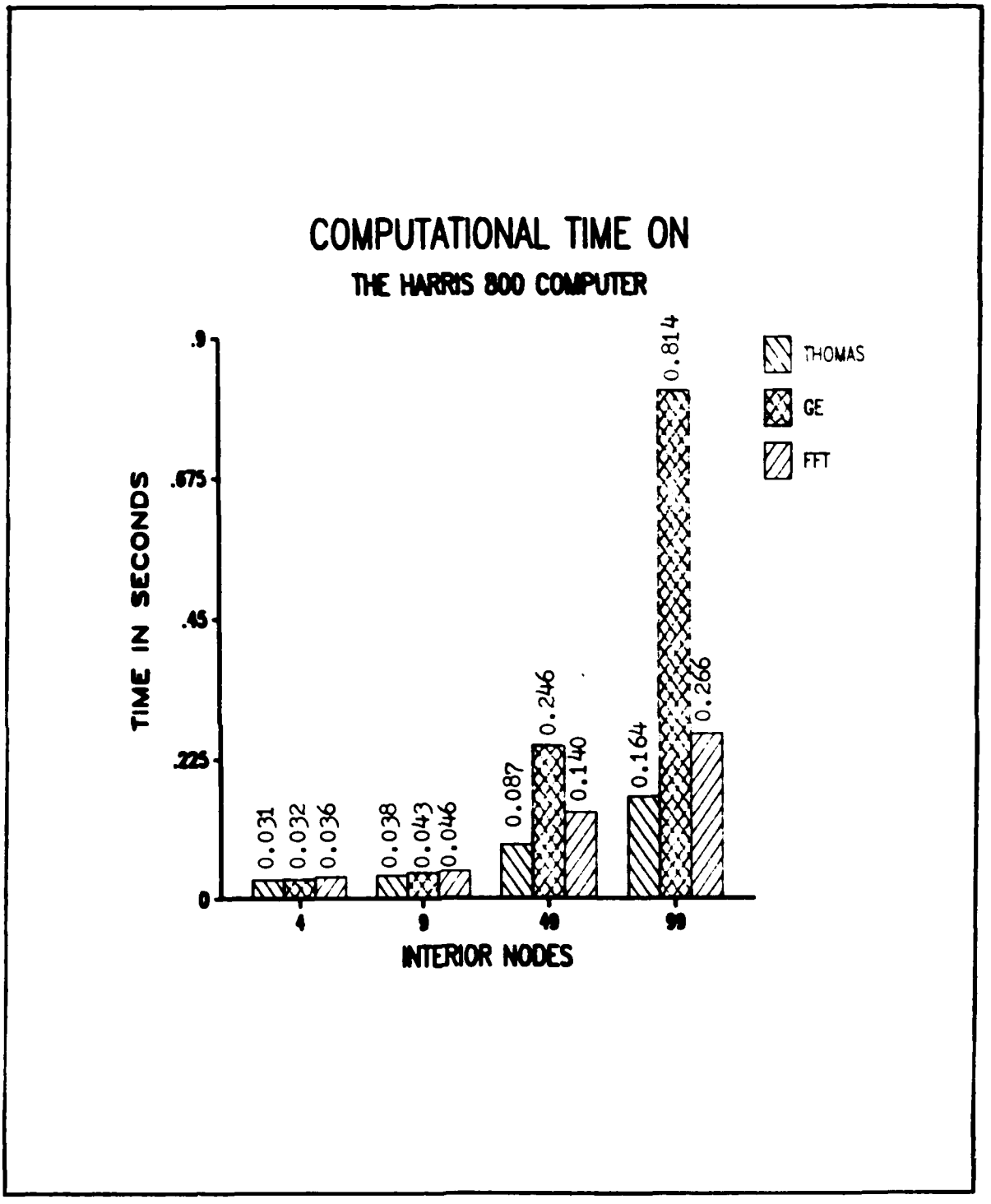


Figure 8. Comparison of Total Computer Time for the Thomas Method, the GE Method, and the FFT Method

IV. Poisson's Equation in Two Dimensions

The second problem examined in this study is Poisson's equation in two dimensions. The form of the equation is

$$\nabla^2 U(x,y) = -2 \qquad (4.1)$$

with boundary conditions

$$U(0,y) = 0 \qquad (4.2\text{-a})$$

$$U(8,y) = 0 \qquad (4.2\text{-b})$$

$$U(x,0) = 0 \qquad (4.2\text{-c})$$

$$U(x,6) = 0 \qquad (4.2\text{-d})$$

where U(x,y) is the unknown function to be determined. This BVP is described by equations (4.1) and (4.2) as a rectangle with dimensions of 8 in the x-direction and 6 in the y-direction (see Figure 9).

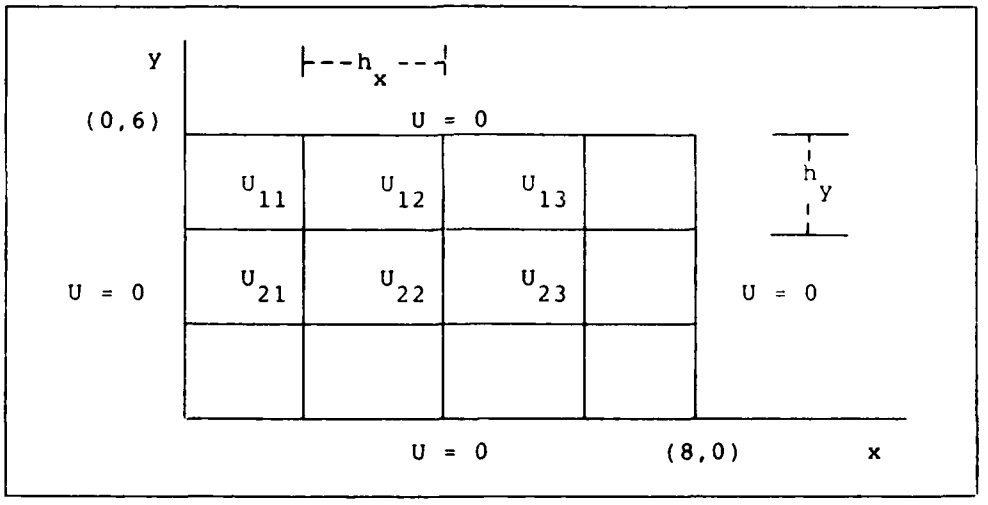


Figure 9. 2-D BVP with 6 Interior Nodes

TABLE II

Exact Solution to 2-D Poisson's Equation
at 6 Interior Nodes

Nodal Point	Exact Solution
U_{11}	4.18250
U_{12}	5.95685
U_{13}	4.18250
U_{21}	4.18250
U_{22}	5.95685
U_{23}	4.18250

Analytical Solution

Equation (4.1) can be solved using variable substitution, separation of variables, and Fourier Series techniques. The complete solution to this BVP can be found in Appendix C. The solution to equation (4.1), after applying the boundary conditions in equation (4.2) is

$$U(x,y) = (8x - x^2 - 512/\pi^3)$$

$$\sum_{n=1}^{\infty} \frac{\{\sinh[(2n-1)\pi y/8]+\sinh[(2n-1)\pi(6-y)/8]\}}{(2n-1)^3\sinh[3(2n-1)\pi/4]}$$

$$\sin[(2n-1)\pi x/8] \qquad (4.3)$$

The exact solutions for U(x,y) at the 6 interior nodes shown in Figure 9 are found by solving equation (4.3) at each nodal point. A simple BASIC program was written to solve for these nodal points. The solutions converged to the fifth decimal place after only six terms were summed. A summary of the exact solutions are in Table II.

Numerical Solution

The equation (4.1) can also be found using numerical approximations as outlined in Chapter II. The mesh is superimposed over the rectangle as shown in Figure 9. There are n equally spaced nodes in the x-direction, with a mesh-size of h_x, and m equally spaced nodes in the y-direction, with a mesh-size of h_y. The total number of interior nodes is equal to m x n, which is equal to N. The size of h_x and h_y will dictate the number of n and m points, hence, the number of simultaneous equations necessary to be solved. The accuracy of the solution in general is dependent on the value of n and m as in the one dimensional problem.

Finite Difference Method. By allowing $h_x=h_y$ and representing them by h, the solution with FDM, using central difference approximations, can be simplified. Equation (4.1) can be approximated by

$$\frac{\delta^2 U}{h_x^2} + \frac{\delta^2 U}{h_y^2} + 2 = 0 \qquad (4.4)$$

since $h_x=h_y=h$, then

$$1/h^2[U_{j+1,k}-2U_{j,k}+U_{j-1,k}+U_{j,1+k}-2U_{j,k}+U_{j,1-k}]+2 = 0 \qquad (4.5)$$

which can be simplified to

$$1/h^2[U_{j+1,k}+U_{j-1,k}+U_{j,1+k}+U_{j,1-k}-4U_{j,k}] + 2 = 0 \qquad (4.6)$$

Equation (4.6) is a matrix equation which can be solved directly by the Gauss Elimination method. Figure 10 is an example of equation (4.6) with 6 interior nodes (3 nodes in the x-direction and 2 nodes in the y-direction).

$$\begin{bmatrix} -4 & 1 & 0 & 1 & 0 & 0 \\ 1 & -4 & 1 & 0 & 1 & 0 \\ 0 & 1 & -4 & 0 & 0 & 1 \\ 1 & 0 & 0 & -4 & 1 & 0 \\ 0 & 1 & 0 & 1 & -4 & 1 \\ 0 & 0 & 1 & 0 & 1 & -4 \end{bmatrix} \begin{bmatrix} U_{11} \\ U_{12} \\ U_{13} \\ U_{21} \\ U_{22} \\ U_{23} \end{bmatrix} = \begin{bmatrix} -8 \\ -8 \\ -8 \\ -8 \\ -8 \\ -8 \end{bmatrix}$$

Figure 10. 2-D Solution to Equation (4.6) with 6 Interior Nodes

Fast Fourier Transform Method

The solution to equation (4.1) using the FFT method is quite similar to the method described in Chapter III for the one dimensional problem except an additional dimension is added. By allowing U(x,y) and F(x,y)=2 to be extended as odd, 2L-periodic functions in both x and y, then equation (4.1) can be solved using numerical approximations in three steps (16:151). The derivation of each of these steps can be found in Appendix D. The first step is to compute the coefficients, F_{jk}, by the use of the Fast Fourier transformation of

$$F_{jk} = \frac{4(h_x h_y)}{x_{max} y_{max}} \sum_{m=1}^{M} \sin(j\pi m\Delta x/x_{max}) \sum_{n=1}^{N} \sin(k\pi n\Delta y/y_{max}) f_{mn} \qquad (4.7)$$

where x_{max} is the maximum value of x in the x-direction and y_{max} is the maximum value of y in the y-direction. The value, f_{mn}, is defined as the discrete value of F(x,y), where $h_x = x_{max}/(M+1)$, the mesh size in the

x-direction, and $h_y = y_{max}/(N+1)$, the mesh size in the y-direction. M is defined as the number of mesh points in the x-direction and N is defined as the number of mesh points in the y-direction. The second step is to compute the coefficients, C_{jk}, by applying

$$C_{jk} = \frac{-F_{jk}}{-[(j\pi/x_{max})^2 + (k\pi/y_{max})^2]} \tag{4.8}$$

The final step is to compute the nodal values of the function U(x,y) by applying the inverse FFT from

$$U(x,y) = \sum_{j=1}^{M-1} \sum_{k=1}^{N-1} C_{jk} \sin(j\pi x/x_{max}) \sin(k\pi y/y_{max}) \tag{4.9}$$

Computer Analysis

The FORTRAN code for the FFT two dimensional problem is listed in Appendix A, and uses the IMSL Routine FFTCC. The Gauss Elimination codes were developed using the IMSL Routine LEQT1F. Because of the straight forward nature of the LEQT1F subroutine the FORTRAN codes for the GE method are not listed in an appendix.

FFT Algorithm. There are three important points that must be considered when applying the two dimensional FFT algorithm. The first two points are the same as outlined in Chapter III under the FFT Algorithm subheading. First, the sampled interval must be extended over 2L to account for the FFTCC subroutine that makes computations over a 2L interval. Second, the DC component must again be removed, as in the one dimensional problem, prior to applying step two (solving for C_{jk} coefficients). This

is accomplished by starting to sum at j=2 and k=2 rather than j=1 and k=1 . Finally, care must be taken as to which FFT algorithm is implemented. Ideally, an algorithm that computes sine FFTs and inverse sine FFTs would be desired. The author used the IMSL routine FFTCC, for the two dimensional problem, which computes the FFT of a complex array in one dimension. This was necessary because a sine FFT that computed both the since FFT and the inverse sine FFT was not available. The FFTCC subroutine uses a complex kernel, which means that both the real and the imaginary part of the function is computed. In order to compute the two dimensional FFT of a sine function it was necessary to write a subroutine to compute the two dimensional FFT and eliminate the real portion of each computation. Appendix E contains a mathematical explanation as to the reasoning behind this computation. With the cosine terms removed the transform becomes a double sine series as shown by equations (4.7) and (4.8). The number of computations required to calculate the two dimensional FFT then becomes $8NLOG_2N + 10N$ (11:217).

Exact Solution. The exact solution to equation (4.1) was found at six points as illustrated in Figure 9. The value of each nodal point is listed in Table II. These exact solutions were then compared to the solutions obtained from using the FDM (Gauss Elimination) and the FFT method.

Nodal Points. Equation (4.1) was solved numerically with three increasingly larger number of nodal points. The first mesh size included 35 interior nodes, which resulted in the h_x and h_y being equal to 1.0. The next mesh size included 165 interior nodes, which resulted in the h_x and h_y being equal to 0.5. The final mesh size included 713 interior nodes,

which resulted in the h_x and h_y being equal to 0.25. Appendix B contains the results of these computations.

Average Error. The accuracy of the approximations was determined using the same average percent error equation as discussed in Chapter III.

$$\langle E \rangle = \left| \frac{\text{Exact Solution - Numerical Solution}}{\text{Exact Solution}} \right| \times 100 \qquad (3.19)$$

Comparison of Approximations to Exact Solutions. The plots of the average percent error to each set of interior nodes are found in Figure 11. As the number of interior nodes increases the accuracy of the FFT method increases. Note, that the accuracy of the FDM decreases as the number of nodes increases. This decrease in accuracy is due to roundoff error (6:38). Table IX, Appendix B, contains the values of the analytical solutions as compared to the GE and FFT methods.

Computational Time. The comparison of the computational speed between the GE method and the FFT method can be found in Tables X and XI, Appendix B. Table X contains the values of the algorithm computational time required to compute either the GE or the FFT. Table XI contains the values of the total computer time used to run each program. Figures 12 and 13 provide a graphic illustration of the computational times. Figure 12 is the comparison of the algorithm computations, whereas Figure 13 is the comparison of the total computer time. It is obvious that the N^3 operations required for GE method really become significant when the interior nodes are increased to 165, as compared to the FFT, $8NLOG_2N + 10N$.

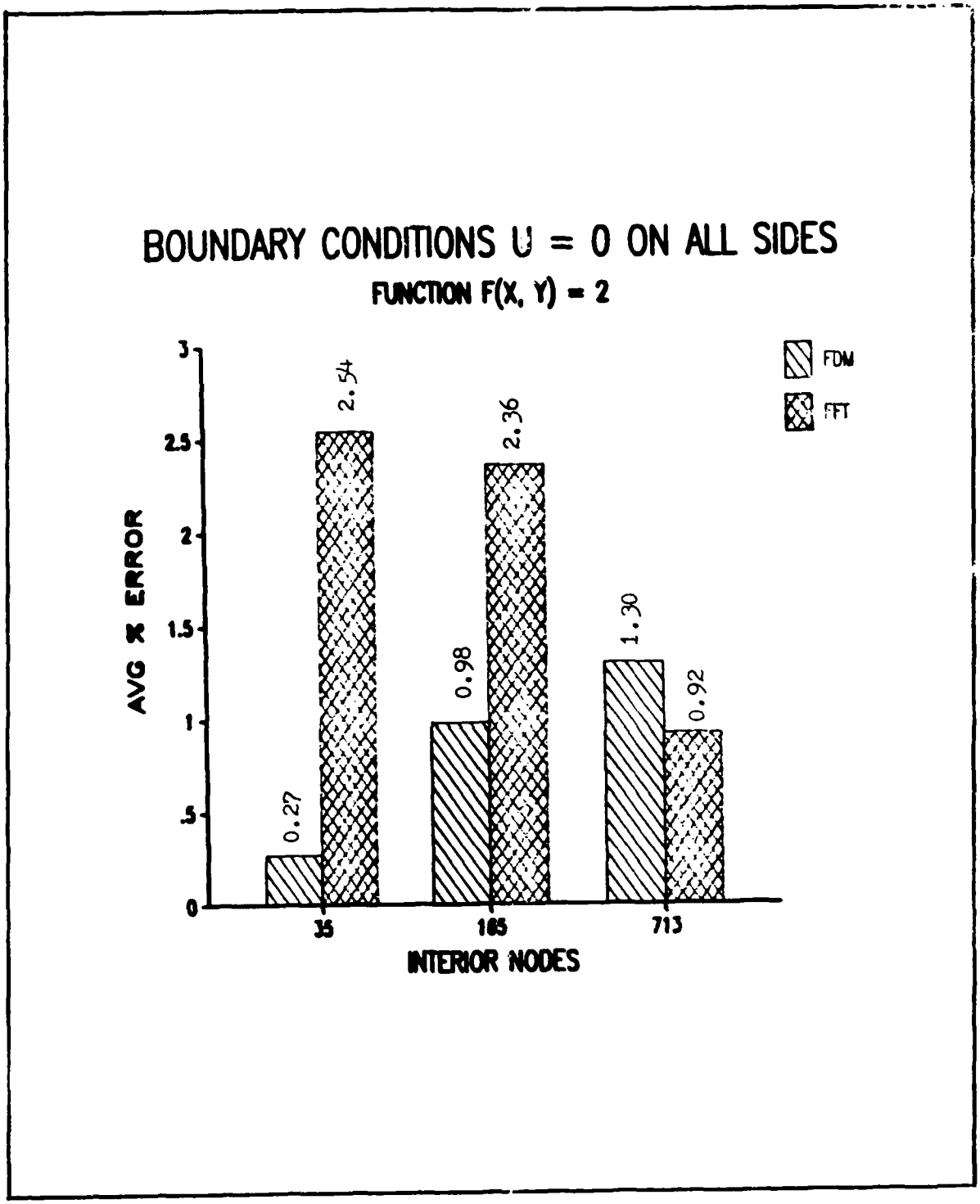


Figure 11. Average Error Comparison Between the FDM, FFT Method, and the Analytical Solution in Two Dimensions

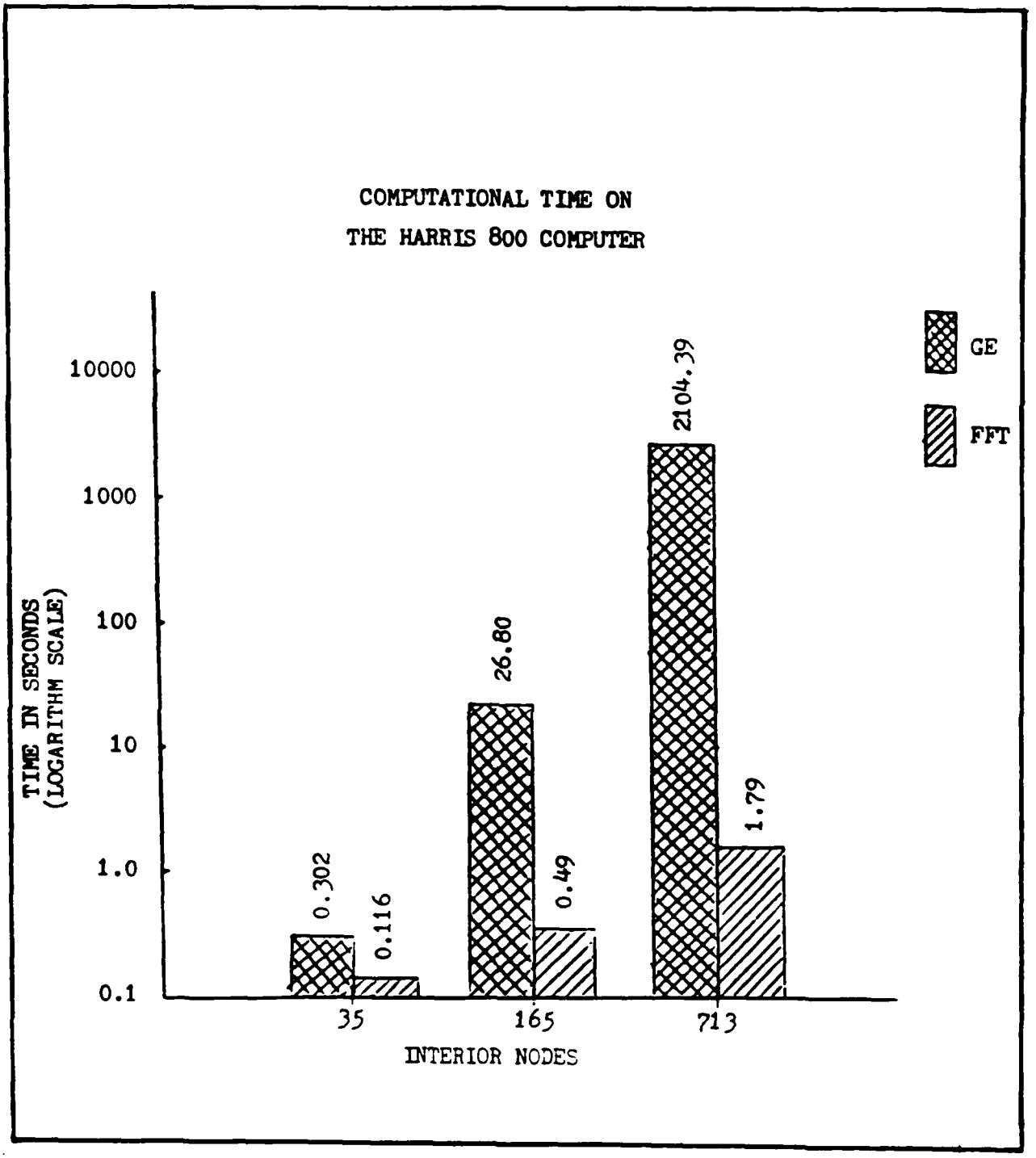


Figure 12. Comparison of Algorithm Computational Times for the GE and FFT Methods in the Two Dimensional BVP

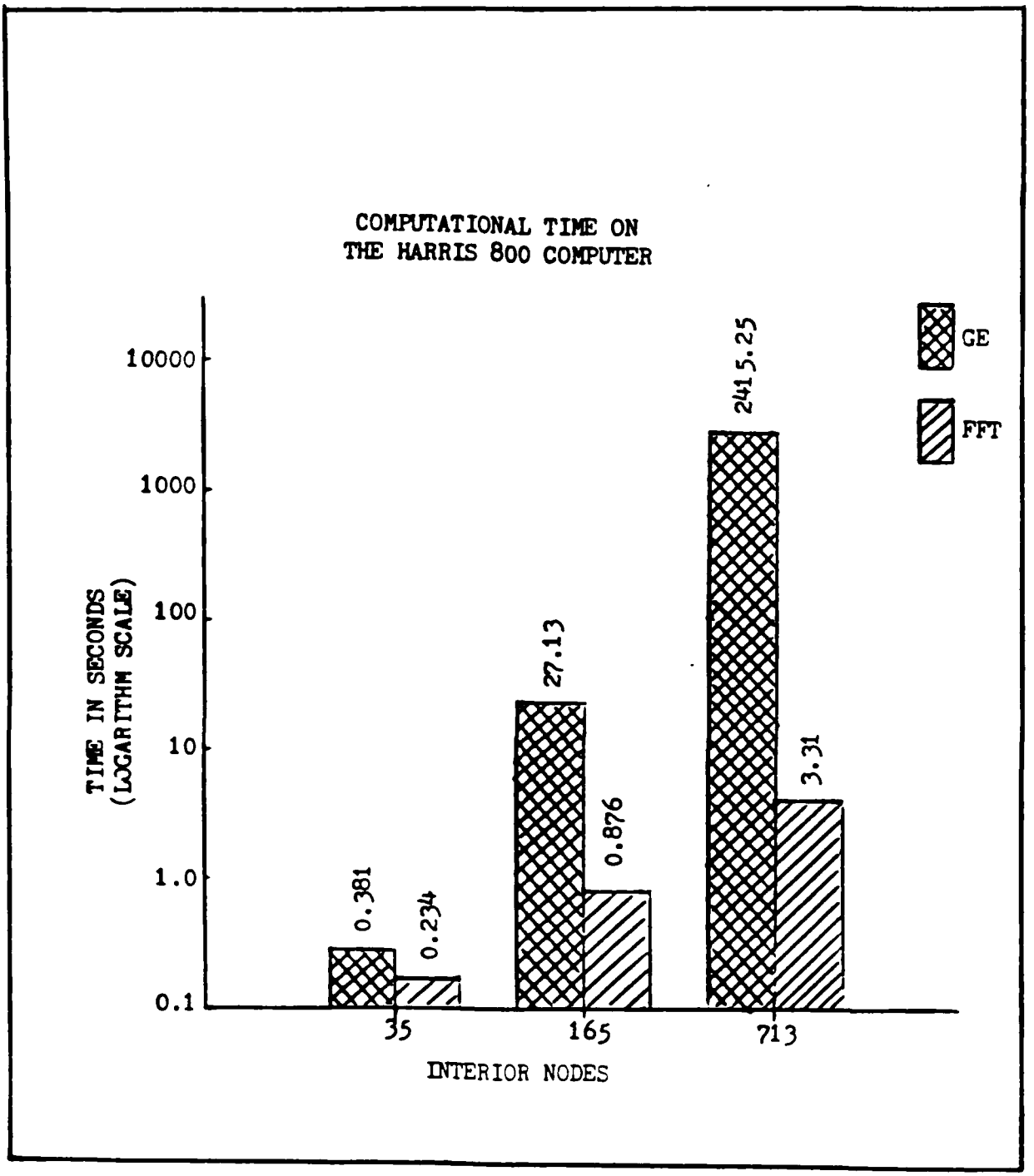


Figure 13. Comparison of Total Computer Time for the GE and FFT Methods in the Two Dimensional BVP

Not only does the computational time increase for the GE method, but the computer storage space increases dramatically. For example, to compute the 713 interior nodes for the GE method, it requires a matrix 713x713 plus two one dimensional matrices of 713. This equates to over 500,000 memory storage locations (10). On the other hand, to compute the same 713 interior nodes using the FFT method, it requires two 75x82 matrices, one 120x160 matrix, one 160 matrix, and one complex 120x160 matrix. This equates to just over 50,000 storage locations. (The FFT matrices are computing 4x713 interior nodes, because the FFT method requires a 2L interval in both the x-direction and the y-direction.) This large storage requirement, in addition to the computational time, detract from the efficiency of the GE method and enhance the FFT method.

V. Conclusions and Recommendations

Conclusions

The results from the one dimensional problem show that the Thomas method (Tridiagonal Method) is the most efficient of the methods examined because it provides the exact solution, and the computational time requires only 8N operations (3:48). The FFT method, though faster than the GE method as the value of N increases, cannot provide the exact solution. The GE method, as does the Thomas method, provides the exact solution with only 4 nodal points. This means small memory storage requirements, and small computer time usage in the one dimensional BVP. The exact solution using the FDM is because the FDM is based on a polynomial approximation, whereas the FFT is based on a trigonometric approximation.

The power of the FFT method is not realized until the Poisson's equation is analyzed in two dimensions. The Thomas method cannot be utilized in the two dimensional problem, because a tridiagonal cannot be formed in two dimensions when the matrix is full (5:56). The Thomas method is used in the Hockney method for two dimensions, where the two dimensional problem is broken down into a one dimensional FFT and a one dimensional Tridiagonal (8:95; 16:151). Additionally the Thomas method can be used in two dimensions if the matrix is sparsely populated (5:56). In the case of a fully populated matrix the comparison of direct numerical methods is narrowed to only the GE method and the FFT method.

The average percent error of the FFT method in the two dimensional BVP decreases as the number of nodal points is increased, as shown in

Figure 11. The average percent error of the GE method increases with increased nodal points due to roundoff error (6:38). The best percent error with the GE method is 0.27% at 35 nodal points, whereas the best percent error with the FFT method is 0.92% at 713 nodal points. One would reason then that the Ge method is more efficient since it provides greater accuracy with less nodal points. This is true, but when a larger set of nodal points are required, which is often the case in engineering, the value of the GE method is greatly reduced. This is apparent when the computational times and the computer storage requirements of the GE method are compared to the FFT method (see Figures 12 and 13). The FFT method is 3 times faster with 35 nodal points than the GE method, 55 times faster with 165 nodal points, and 1175 times faster with 713 nodal points (see Tables X and XI, Appendix B). Additionally, the computer storage requirement for the GE method is 10 times greater than the FFT method as discussed in Chapter IV. Hence, not only is the FFT method faster than the GE method, but it requires less computer memory. The FFT method seems to provide the most efficient two dimensional method of solving Poisson's equation numerically with full matrices. The ADI method is faster than the FFT in two dimensions, but is only good for sparse matrices (5:56). If the two dimensional BVP is expanded to three dimensions, then the efficiency of the FFT method could prove to be even greater.

In the one dimensional problem four cases were considered that varied both the value of F(x) and the boundary conditions. One question of great concern is whether or not the FFT method can accept boundary conditions other than zero (13:697; 16:89). This study found that the FFT method can be used with other than zero boundary conditions in the BVP. This procedure

was discussed in Chapter III under the Fast Fourier Transform Method. The two dimensional Poisson equation was only studied with boundary conditions equal to zero. It is logical to assume that non-zero boundary conditions can be applied to the two dimensional BVP and solved using the same methods employed in the zero boundary condition problem. The difficulty associated with this would be to describe the input function as was discussed in Chapter III. The same procedure would be used as was done in the one dimensional problem, except the function would be expanded to two dimensions.

Recommendations

This study has only touched the surface on how the FFT method can be used to solve BVPs. Only boundary conditions of zero and constants were used in this study to establish a basis for the validity of this method. The FFT method needs to be expanded to non-zero boundary conditions in two dimensions. (J. Rosser, from the University of Wisconsin, has done some work in the area (12:38).) And, then expanded to three dimensions as mentioned by Fox and Otto (5:56). Additionally, there has been limited work in the use of Neumann boundary conditions (12:41; 13:707). Additional research needs to be done on how extensive FFTs can be used in conjunction with Neumann conditions. In this study and in all the references found on the FFT method, the only equation studied was the Poisson equation. Hence, there is a question as to the ability of this FFT method to provide numerical solutions for equations other than elliptic equations. Research needs to be done to determine the limitations of FFTs on solving BVP

with parabolic and hyperbolic equations. Another recommendation is to develop a FORTRAN code that will compute the sine FFT and the inverse sine FFT, thus increasing the efficiency of the FFT algorithm. The power of the FFT is extraordinary and needs to be examined extensively beyond the scope of this study.

Appendix A: Computer Codes

This Appendix contains the listing of the FORTRAN programs used in the one and two dimensional cases for computing the FFT. All FFT FORTRAN programs were run on the Harris 800 computer, using the IMSL routine FFTCC.

The following are the titles of the programs in Appendix A and their function.

TITLE	FUNCTION
FFT	Compute one dimensional Poisson equation with F(x)=40 and Boundary Conditions equal to zero using the FFT algorithm.
NFFT	Compute one dimensional Poisson equation with F(x)=40 and Boundary Conditions U(0)=2 and U(10)=8 using the FFT algorithm.
FFT2	Compute one dimensional Poisson equation with F(x)=40 and Boundary Conditions equal to zero using the FFT algorithm.
NFFT2	Compute one dimensional Poisson equation with F(x)=40 and Boundary Conditions U(0)=2 and U(10)=8 using the FFT algorithm.
FFT2D1	Compute the two dimensional Poisson equation with F(x,y)=2 and Boundary Conditions equal to zero using the FFT algorithm.

```
*PROGRAM NAME       FFT     F(X) = 40  (REC FUNCTION)
*AUTHOR             TODD R JONES
*DATE               14 OCT 1985
*************************************************************
*THIS PROGRAM WILL PROVIDE A NUMERICAL SOLUTION TO A       *
*ONE DIMENSIONAL BVP USING FFT.  THE BOUNDARIES MUST BE    *
*ZERO AT BOTH ENDS.  THE FUNCTION F(X) MUST BE KNOWN AND   *
*PLACED IN THE PROGRAM.  THE FFT SUBROUTINE IS AN IMSL     *
*SUBROUTINE CALLED FFTCC AND WILL TRANSFORM 300 VALUES     *
*OF N.                                                     *
*************************************************************
*
*    DECLARATION OF VARIABLES
*
      INTEGER N,IWK(1050)
      REAL WK(1050),L(300)
      COMPLEX A(300)
      PI = 4.*ATAN(1.)
*
*   INPUT NUMBER OF POINTS TO BE TRANSFORMED
*
      PRINT*,'ENTER NUMBER OF DATA POINTS TO BE TRANSFORMED'
      READ*,N
*
*   INPUT FUNTION DESCRIPTION F(X)
*
      A(1) = (0,0)
      A(N/2) = (0,0)
      A(N) = (0,0)
      A(2) = (40,0)
      A(N/2+1) = (-40,0)
      DO 10 I = 3,N/2-1
        A(I) = A(2)
10    CONTINUE
      DO 20 I = N/2+1,N-1
        A(I) = A(N/2+1)
20    CONTINUE
*
* START TIMER
*
      CALL BTIME
      DO 30 I = 1,N
        A(I) = CONJG(A(I))
30    CONTINUE
*
* USE FFT SUBROUTINE FFTCC
*
      CALL FFTCC(A,N,IWK,WK)
      DO 40 I = 1,N
        A(I) = CONJG(A(I))*2/N
40    CONTINUE
      DO 50 I = 2,N
        A(I) = ((10/((I-1)*PI))**2)*A(I)
50    CONTINUE
```

```
      CALL FFTCC(A,N,IWK,WK)
      CALL ETIME
******** STOP TIMER ********
*
*   PRINT VALUES OF U(N)
*
      OPEN(1,FILE='DAT1FT1')
      DO 60 I = 1,N
        L(I) = REAL(A(I))
60    CONTINUE
      DO 70 I = 1,N/2
        WRITE (1,100) L(I)
100     FORMAT(' ',F8.3)
70    CONTINUE
      CLOSE (1)
      END
```

```
*PROGRAM NAME        NFFT   F(X) = 40 (REC FUNCTION)
*AUTHOR              TODD JONES
*DATE                1 NOV 1985
*****************************************************************
*THIS PROGRAM WILL PROVIDE A NUMERICAL SOLUTION TO A            *
*ONE DIMENSIONAL BVP USING FFT.  THE BOUNDARIES MUST BE         *
*U(0)=2, AND U(10)=8.  THE FUNCTION F(X) MUST BE KNOWN AND      *
*PLACED IN THE PROGRAM.  THE FFT SUBROUTINE IS AN IMSL          *
*SUBROUTINE CALLED FFTCC AND WILL TRANSFORM 300 VALUES          *
*OF N.                                                          *
*****************************************************************
*
*    DECLARATION OF VARIABLES
*
      INTEGER N,IWK(1050)
      REAL WK(1050),L(300)
      COMPLEX A(300)
      PI = 4.*ATAN(1.)
*
*   INPUT NUMBER OF POINTS TO BE TRANSFORMED
*
      PRINT*,'ENTER NUMBER OF DATA POINTS TO BE TRANSFORMED'
      READ*,N
*
* INPUT FUNCTION DESCRIPTION F(X)
*
      A(1) = (2.0,0)
      A(2) = (40.0,0)
      A(N/2) = (8.0,0)
      A(N/2+1) = (-2.0,0)
      A(N/2+2) = (-40.0,0)
      A(N) = (-8.0,0)
      DO 10 I = 3,N/2-1
        A(I) = A(2)
10    CONTINUE
      DO 20 I = N/2+2,N-1
        A(I) = A(N/2+2)
20    CONTINUE
*
* START TIMER
*
      CALL BTIME
      DO 30 I = 1,N
        A(I) = CONJG(A(I))
30    CONTINUE
*
* USE FFT SUBROUTINE FFTCC
*
      CALL FFTCC(A,N,IWK,WK)
      DO 40 I = 1,N
        A(I) = CONJG(A(I))*2/N
40    CONTINUE
      DO 50 I = 2,N
        A(I) = ((10/((I-1)*PI))**2)*A(I)
```

```
50    CONTINUE
      CALL FFTCC(A,N,IWK,WK)
      CALL ETIME
******** STOP TIMER ********
*
*   PRINT VALUES OF U(N)
*
      OPEN(1,FILE='DAT1NFT1')
      DO 60 I = 1,N
        L(I) = REAL(A(I))
      PRINT*,I,' ',L(I)
60    CONTINUE
      DO 70 I = 1,N/2
        WRITE(1,100) L(I)
100     FORMAT(' ',F8.3)
70    CONTINUE
      CLOSE (1)
      END
```

```
*PROGRAM NAME       FFT2     F(X) = X  (RAMP FUNCTION)
*AUTHOR             TODD R JONES
*DATE               14 OCT 1985
*************************************************************
*THIS PROGRAM WILL PROVIDE A NUMERICAL SOLUTION TO A       *
*ONE DIMENSIONAL BVP USING FFT.  THE BOUNDARIES MUST BE    *
*ZERO AT BOTH ENDS.  THE FUNCTION F(X) MUST BE KNOWN AND   *
*PLACED IN THE PROGRAM.  THE FFT SUBROUTINE IS AN IMSL     *
*SUBROUTINE CALLED FFTCC AND WILL TRANSFORM 300 VALUES     *
*OF N.                                                     *
*************************************************************
*
*   DECLARATION OF VARIABLES
*
      INTEGER N,IWK(1050)
      REAL WK(1050),L(300)
      COMPLEX A(300)
      PI = 4.*ATAN(1.)
*
*  INPUT NUMBER OF POINTS TO BE TRANSFORMED
*
      PRINT*,'ENTER NUMBER OF DATA POINTS TO BE TRANSFORMED'
      READ*,N
*
*  INPUT FUNTION DESCRIPTION F(X)
*
      A(0) = (0,0)
      A(1) = -10
      A(N/2) = (0,0)
      A(N) = (0,0)
      DO 10 I = 2,N/2-1
        A(I) = A(I-1) + 20./N
10    CONTINUE
      DO 20 I = N/2+1,N-1
        A(I) = A(I-1) + 20./N
20    CONTINUE
*
* START TIMER
*
      CALL BTIME
      DO 30 I = 1,N
        A(I) = CONJG(A(I))
30    CONTINUE
*
* USE FFT SUBROUTINE FFTCC
*
      CALL FFTCC(A,N,IWK,WK)
      DO 40 I = 1,N
        A(I) = CONJG(A(I))*2/N
40    CONTINUE
      DO 50 I = 2,N
        A(I) = ((10/((I-1)*PI))**2)*A(I)
50    CONTINUE
      CALL FFTCC(A,N,IWK,WK)
```

```
      CALL ETIME
******** STOP TIMER *********
*
*  PRINT VALUES OF U(N)
*
      DO 100 I = 1,N
        L(I) = REAL(A(I))
        PRINT*,L(I)
100   CONTINUE
      END
```

```
*PROGRAM NAME      NFFT2   F(X) = X  (RAMP FUNCTION)
*AUTHOR            TODD JONES
*DATE              6 NOV 1985
*****************************************************************
*THIS PROGRAM WILL PROVIDE A NUMERICAL SOLUTION TO A            *
*ONE DIMENSIONAL BVP USING FFT.  THE BOUNDARIES MUST BE         *
*U(0)=2, AND U(10)=8. THE FUNCTION F(X) MUST BE KNOWN AND       *
*PLACED IN THE PROGRAM.  THE FFT SUBROUTINE IS AN IMSL          *
*SUBROUTINE CALLED FFTCC AND WILL TRANSFORM 300 VALUES          *
*OF N.                                                          *
*****************************************************************
*
*    DECLARATION OF VARIABLES
*
      INTEGER N,IWK(1050)
      REAL WK(1050),L(300)
      COMPLEX A(300)
      PI = 4.*ATAN(1.)
*
*   INPUT NUMBER OF POINTS TO BE TRANSFORMED
*
      PRINT*,'ENTER NUMBER OF DATA POINTS TO BE TRANSFORMED'
      READ*,N
      PRINT*,'ENTER VALUE FOR STEP INCREMENT "Z"'
      READ*,Z
      PRINT*,'ENTER VALUE FOR A(N/2+1)'
      READ*,A(N/2+2)
*
*   INPUT FUNTION DESCRIPTION F(X)
*
      A(1) = (2.0,0)
      A(N/2) = (8.0,0)
      A(N) = (-2.0,0)
      A(N/2+1) = (-8.0,0)
      DO 10 I = 2,N/2-1
        A(I) = A(I-1) + Z
10    CONTINUE
      DO 20 I = N/2+2,N-1
        A(I) = A(I-1) + Z
20    CONTINUE
*
* START TIMER
*
      CALL BTIME
      DO 30 I = 1,N
         A(I) = CONJG(A(I))
30    CONTINUE
*
* USE FFT SUBROUTINE FFTCC
*
      CALL FFTCC(A,N,IWK,WK)
      DO 40 I = 1,N
         A(I) = CONJG(A(I))*2/N
40    CONTINUE
```

```
      DO 50 I = 2,N
        A(I) = ((10/((I-1)*PI))**2)*A(I)
50    CONTINUE
      CALL FFTCC(A,N,IWK,WK)
      CALL ETIME
******** STOP TIMER *********
*
*  PRINT VALUES OF U(N)
*
      DO 100 I = 1,N
        L(I) = REAL(A(I))
        PRINT*,I,' ',L(I)
100   CONTINUE
      OPEN (1,FILE='DAT1NF2')
      DO 60 I = 1,N/2
        WRITE(1,200) L(I)
200     FORMAT(' ',F8.4)
60    CONTINUE
      CLOSE (1)
      END
```

```
*PROGRAM NAME        FFT2D1  F(X,Y)= 2  (REC FUNCTION)
*AUTHOR              TODD JONES
*DATE                23 OCT 85
*****************************************************************
*THIS PROGRAM WILL PROVIDE A NUMERICAL SOLUTION TO A TWO (2)
*DIMENSIONAL BVP (POISSON EQUATION) USING FFTs.  THE
*BOUNDARIES MUST BE ZERO ON ALL SIDES.  THE FFT
*SUBROUTINE IS AN IMSL SUBROUTINE CALLED FFTCC.
*****************************************************************
*
* DECLARATION OF VARIABLES
*
      INTEGER IWK(75,82),N,IA,IJOB
      REAL RWK(75,82),L(120,160)
      COMPLEX A(120,160), CWK(160)
      COMMON IWK,RWK,CWK,L,A
      PI = 4.0 * ATAN(1.)
*
* INPUT NUMBER OF POINTS TO BE TRANSFORMED
*
      PRINT*,'ENTER NO. OF POINTS TO BE TRANSFORMED-X DIR'
      READ*,N1
      PRINT*,'ENTER NO. OF POINTS TO BE TRANSFORMED-Y DIR'
      READ*,N2
      IA1 = N1
      IA2 = N2
*
* ENTER COMPLEX ARRAY F(X)
*
***** QUADRANT I *****
      DO 10 I = 2,N1/2
      DO 10 J = 2,N2/2
          A(I,J) = (2.0,0.0)
10    CONTINUE
***** QUADRANT II *****
      DO 15 I = N1/2+1,N1-1
      DO 15 J = 2,N2/2
          A(I,J) = (-2.0,0.0)
15    CONTINUE
***** QUADRANT III *****
      DO 20 I = 2,N1/2
      DO 20 J = N2/2+1,N2-1
          A(I,J) = (-2.0,0.0)
20    CONTINUE
***** QUADRANT IV *****
      DO 25 I = N1/2+1,N1-1
      DO 25 J = N2/2+1,N2-1
         A(I,J) = (2.0,0.0)
25    CONTINUE
*
* START TIMER
*
      CALL BTIME
*
```

```
*  SET IJOB
*
      IJOB = -1
*
*  USE IMSL ROUTINE FFT3D
*
      CALL TWODIM(A,IA1,IA2,N1,N2,IJOB,IWK,RWK,CWK)
      DO 30 I = 2,N1
      DO 30 J = 2,N2
        A(I,J) = (4*A(I,J))/((PI*(J-1)/X)**2+(PI*(I-1)/Y)**2)
30    CONTINUE
      IJOB = +1
      CALL TWODIM(A,IA1,IA2,N1,N2,IJOB,IWK,RWK,CWK)
      CALL ETIME
******** STOP TIMER ********
      DO 35 I = 1,N1
      DO 35 J = 1,N2
        L(I,J) = REAL(A(I,J))
35    CONTINUE
*
*  PRINT VALUES OF NODAL POINTS TO A FILE
*
      OPEN(1,FILE='DFT2D1')
      DO 40 I = 1,N1/2
      DO 40 J = 1,N2/2
        WRITE(1,100) I,J,L(I,J)
100     FORMAT(' ',I4,I4,F8.3)
40    CONTINUE
      CLOSE (1)
*
*  PRINT ARRAYS TO SCREEN
*
      DO 45 I = 1,N1/2
      DO 45 J = 1,N2/2
        PRINT 200,I,J,L(I,J)
200     FORMAT(' ',I4,I4,F8.3)
45    CONTINUE
      END
*****************************************************************
*****************************************************************
******************** SUBROUTINE TWODIM **************************
*****************************************************************
*****************************************************************
      SUBROUTINE TWODIM(A,IA1,IA2,N1,N2,IJOB,IWK,RWK,CWK)
      INTEGER IA1,IA2,N1,N2,IJOB,IWK(1)
      REAL RWK(1)
      COMPLEX A(IA1,IA2),CWK(1)
      INTEGER I,J,K,L,M,N
      REAL R12
      COMPLEX C12
*
*  DETERMINE TRANSFORM OR INVERSE TRANSFORM
*
      IF(IJOB.GT.0) GO TO 10
```

```
*
* INVERSE TRANSFORM
*
      DO 15 I = 1,N1
      DO 15 J = 1,N2
        A(I,J) = CONJG(A(I,J))
15    CONTINUE
*
* TRANSFORM SECOND SUBSCRIPT
*
10    DO 20 L = 1,N1
        DO 25 M = 1,N2
          CWK(M) = A(L,M)
25      CONTINUE
        CALL FFTCC(CWK,N2,IWK,RWK)
        DO 30 J = 1,N2
          A(L,J) = AIMAG(CWK(J))
30      CONTINUE
20    CONTINUE
*
* TRANSFORM FIRST SUBSCRIPT
*
      DO 35 J = 1,N2/2
        DO 40 K = 1,N1
          CWK(K) = A(K,J)
40      CONTINUE
        CALL FFTCC(CWK,N1,IWK,RWK)
        DO 45 L = 1,N1
          A(L,J) = AIMAG(CWK(L))
45      CONTINUE
35    CONTINUE
*
* INVERSE TRANSFORM
*
      IF (IJOB.GT.0) GO TO 55
      R12 = N1*N2
      C12 = CMPLX(R12,0.0)
      DO 50 I = 1,N1
      DO 50 J = 1,N2
        A(I,J) = CONJG(A(I,J))/C12
50    CONTINUE
55    RETURN
      END
```

Appendix B: Tables of Data

This Appendix contains the tables of data collected from the one dimensional and the two dimesional Poisson's equation to include Average Errors and Computational Times.

Table III

Average Error for the One Dimensional Poisson Equation with F(x)=40 and U(0)=U(10)=0

4-Interior Nodes

X	Analytical U(x)	FDM U(x)	FFT U(x)	Average Error (%) FDM	Average Error (%) FFT
2	320	320	371.97	0	16.24
4	480	480	468.98	0	2.30
6	480	480	445.84	0	7.12
8	320	360	302.58	0	5.44
		Total Average Error		0	7.77

9-Interior Nodes

X	Analytical U(x)	FDM U(x)	FFT U(x)	Average Error (%) FDM	Average Error (%) FFT
2	320	320	360.71	0	12.72
4	480	480	483.90	0	0.81
6	480	480	468.87	0	2.32
8	320	320	315.64	0	1.36
		Total Average Error		0	4.30

49-Interior Nodes

X	Analytical U(x)	FDM U(x)	FFT U(x)	Average Error (%) FDM	Average Error (%) FFT
2	320	320	331.32	0	3.54
4	480	480	482.85	0	0.59
6	480	480	479.08	0	0.19
8	320	320	320.02	0	0.006
		Total Average Error		0	1.08

99-Interior Nodes

X	Analytical U(x)	FDM U(x)	FFT U(x)	Average Error (%) FDM	Average Error (%) FFT
2	320	320	316.35	0	1.14
4	480	480	479.19	0	0.17
6	480	480	481.23	0	0.25
8	320	320	322.47	0	0.77
		Total Average Error		0	0.58

Table IV

Average Error for One Dimensional Poisson's Equation with F(x)=x and U(0)=U(10)=0

4-Interior Nodes

X	Analytical U(x)	FDM U(x)	FFT U(x)	Average Error (%) FDM	Average Error (%) FFT
2	32	32	37.78	0	18.06
4	56	56	64.54	0	15.25
6	64	64	72.12	0	12.68
8	48	48	52.58	0	9.54
		Total Average Error		0	13.88

9-Interior Nodes

X	Analytical U(x)	FDM U(x)	FFT U(x)	Average Error (%) FDM	Average Error (%) FFT
2	32	32	34.97	0	9.28
4	56	56	60.66	0	8.32
6	64	64	68.55	0	7.11
8	48	48	50.65	0	5.52
		Total Average Error		0	7.56

49-Interior Nodes

X	Analytical U(x)	FDM U(x)	FFT U(x)	Average Error (%) FDM	Average Error (%) FFT
2	32	32	32.60	0	1.87
4	56	56	56.98	0	1.75
6	64	64	64.98	0	1.53
8	48	48	48.58	0	1.21
		Total Average Error		0	1.59

99-Interior Nodes

X	Analytical U(x)	FDM U(x)	FFT U(x)	Average Error (%) FDM	Average Error (%) FFT
2	32	32	32.3	0	0.94
4	56	56	56.49	0	0.87
6	64	64	64.49	0	0.76
8	48	48	48.29	0	0.60
		Total Average Error		0	0.78

Table V

Average Error for One Dimensional Poisson's Equation with F(x)=40 and U(0)=2 and U(10)=8

4-Interior Nodes

X	Analytical U(x)	FDM U(x)	FFT U(x)	Average Error (%) FDM	Average Error (%) FFT
2	323.2	323.2	349.56	0	0.40
4	484.8	484.8	441.31	0	9.01
6	485.6	485.6	373.51	0	7.44
8	326.8	326.8	152.10	0	53.45
		Total Average Error		0	17.57

9-Interior Nodes

X	Analytical U(x)	FDM U(x)	FFT U(x)	Average Error (%) FDM	Average Error (%) FFT
2	323.2	323.2	352.78	0	9.15
4	484.8	484.8	478.75	0	1.25
6	485.6	485.6	444.76	0	8.41
8	326.8	326.8	250.86	0	23.23
		Total Average Error		0	10.51

49-Interior Nodes

X	Analytical U(x)	FDM U(x)	FFT U(x)	Average Error (%) FDM	Average Error (%) FFT
2	323.2	323.2	305.35	0	5.52
4	484.8	484.8	474.55	0	2.11
6	485.6	485.6	483.75	0	0.38
8	326.8	326.8	332.95	0	1.88
		Total Average Error		0	2.47

99-Interior Nodes

X	Analytical U(x)	FDM U(x)	FFT U(x)	Average Error (%) FDM	Average Error (%) FFT
2	323.2	323.2	312.89	0	3.19
4	484.8	484.8	477.49	0	1.50
6	485.6	485.6	482.09	0	0.72
8	326.8	326.8	326.69	0	0.03
		Total Average Error		0	1.36

Table VI

Average Error for One Dimensional Poisson's Equation
with F(x)=x and U(0)=2 and U(10)=8

4-Interior Nodes

X	Analytical U(x)	FDM U(x)	FFT U(x)	Average Error (%) FDM	FFT
2	35.2	35.2	42.46	0	20.62
4	60.4	60.4	62.98	0	4.27
6	69.6	69.6	70.56	0	1.38
8	54.8	54.8	59.64	0	8.83
		Total Average Error		0	8.77

9-Interior Nodes

X	Analytical U(x)	FDM U(x)	FFT U(x)	Average Error (%) FDM	FFT
2	35.2	35.2	34.56	0	1.81
4	60.4	60.4	63.78	0	5.59
6	69.6	69.6	66.07	0	5.07
8	54.8	54.8	57.52	0	4.46
		Total Average Error		0	4.23

49-Interior Nodes

X	Analytical U(x)	FDM U(x)	FFT U(x)	Average Error (%) FDM	FFT
2	35.2	35.2	35.63	0	1.22
4	60.4	60.4	59.92	0	0.79
6	69.6	69.6	68.05	0	2.22
8	54.8	54.8	54.53	0	0.49
		Total Average Error		0	1.18

99-Interior Nodes

X	Analytical U(x)	FDM U(x)	FFT U(x)	Average Error (%) FDM	FFT
2	35.2	35.2	35.08	0	0.34
4	60.4	60.4	60.39	0	0.01
6	69.6	69.6	68.10	0	2.15
8	54.8	54.8	54.03	0	1.40
		Total Average Error		0	0.97

Table VII

Computing Time in Seconds for the One Dimensional Poisson Equation[1]

Number of Nodes	Computational Time (Seconds)		
	Thomas	GE	FFT
4	0.001	0.002	0.005
9	0.001	0.007	0.010
49	0.004	0.153	0.056
99	0.008	0.639	0.114

Table VIII

Total Computer Time used in Computing the One Dimensional Poisson Equation[2]

Number of Nodes	Computational Time (Seconds)		
	Thomas	GE	FFT
4	0.031	0.032	0.036
9	0.038	0.043	0.046
49	0.087	0.246	0.140
99	0.164	0.814	0.266

[1]The computing time is only the time used by the computer to compute the particular algorithm.

[2]The total computer time includes the time used by the computer to run the entire computer program.

Table IX

Average Error for the Two Dimensional Poisson Equation with F(x)=2 and all Boundary Conditions equal to Zero

35-INTERIOR NODES

Nodal Points	Analytical U(x,y)	FDM U(x,y)	FFT U(x,y)	Ave Error (%) FDM	Ave Error (%) FFT
U11	4.8125	4.794	4.742	0.38	1.46
U12	4.8125	4.794	4.756	0.38	1.17
U21	5.9568	5.960	6.111	0.05	2.58
U22	5.9568	5.960	6.126	0.05	2.84
U31	4.8125	4.794	4.625	0.38	3.89
U32	4.8125	4.794	4.972	0.38	3.31
		Total Average Error		0.27	2.54

165-INTERIOR NODES

Nodal Points	Analytical U(x,y)	FDM U(x,y)	FFT U(x,y)	Ave Error (%) FDM	Ave Error (%) FFT
U11	4.8125	4.856	4.705	0.90	9.19
U12	4.8125	4.856	4.808	0.90	0.09
U21	5.9568	6.026	5.999	1.16	0.71
U22	5.9568	6.026	6.136	1.16	3.00
U31	4.8125	4.856	4.731	0.90	1.69
U32	4.8125	4.856	4.836	0.90	0.48
		Total Average Error		0.98	2.36

713-INTERIOR NODES

Nodal Points	Analytical U(x,y)	FDM U(x,y)	FFT U(x,y)	Ave Error (%) FDM	Ave Error (%) FFT
U11	4.8125	4.872	4.771	1.23	0.86
U12	4.8125	4.872	4.801	1.23	0.23
U21	5.9568	6.043	6.036	1.44	1.32
U22	5.9568	6.043	6.074	1.44	1.96
U31	4.8125	4.872	4.770	1.23	0.88
U32	4.8125	4.872	4.799	1.23	0.28
		Total Average Error		1.30	0.92

Table X

Computing Times in Seconds for the Two Dimensional Poisson Equation[3]

Number of Nodes	Computational Time (Sec)	
	GE	FFT
35	0.302	0.116
165	26.801	0.490
713	35min 4.393sec	1.790

Table XI

Total Computer Time in Seconds for the Two Dimensional Poisson Equation[4]

Number of Nodes	Computational Time (Sec)	
	GE	FFT
35	0.381	0.234
165	27.134	0.876
713	35min 15.25sec	3.310

[3]The computing time represents only the time used by the computer to compute the particular algorithm.

[4]The total computer time includes the time used by the computer to run the entire computer program.

Appendix C: Analytical Solution to the 2-D BVP

The solution to the BVP described by equation (4.1) and (4.2) is solved by using a variable substitution and the Fourier Series method (4). By making a variable substitution

$$U(x,y) = V(x,y) - x^2 + 8x \tag{C.1-a}$$

then

$$\frac{\partial^2 U}{\partial x^2} = \frac{\partial^2 V}{\partial x^2} - 2 \tag{C.1-b}$$

$$\frac{\partial^2 U}{\partial y^2} = \frac{\partial^2 V}{\partial y^2} \tag{C.1-c}$$

therefore

$$\nabla^2 U = \nabla^2 V - 2 \tag{C.2}$$

so that

$$\nabla^2 V = 0 \tag{C.3}$$

By making the variable substitution in the boundary conditions

$$U(0,y) = V(0,y) = 0 \tag{C.4-a}$$

$$U(8,y) = V(8,y) - 64 + 64 = 0 \tag{C.4-b}$$

$$U(x,0) = V(x,0) - x^2 + 8x = 0 \tag{C.4-c}$$

$$U(x,6) = V(x,6) - x^2 + 8x = 0 \tag{C.4-d}$$

therefore

$$V(0,y) = 0 \tag{C.5-a}$$

$$V(8,y) = 0 \tag{C.5-b}$$

$$V(x,0) = x^2 - 8x \tag{C.5-c}$$

$$V(x,6) = x^2 - 8x \tag{C.5-d}$$

By using separation of variables, equation (C.3) leads to a Sturm-Liouville problem that can be solved using the Fourier series. The solution takes the form

$$V(x,y) = \sum_{n=1}^{\infty} \left[\frac{A_n \sinh(n\pi y/8) + B_n \sinh[(n\pi/8)(6-y)]}{\sinh(3n\pi/4)} \right] \sin(n\pi x/8) \tag{C.6}$$

This can be verified by referring to Churchill and Brown, page 136 (2:136). The coefficients A_n and B_n can be solved by applying boundary conditions from equation (C.5).

$$V(x,y) = \sum_{n=1}^{\infty} B_n \sin(n\pi x/8) = x^2 + 8x \tag{C.7-a}$$

$$V(x,y) = \sum_{n=1}^{\infty} A_n \sin(n\pi x/8) = x^2 + 8x \tag{C.7-b}$$

Thus, $A_n = B_n$ and can be solved as follows.

$$A_n = 2/8 \int_0^8 (x^2-8x)\sin(n\pi x/8)dx \tag{C.8-a}$$

$$A_n = 1/4 \int_0^8 x^2 \sin(n\pi x/8)dx - 2\int_0^8 x\sin(n\pi x/8)dx \tag{C.8-b}$$

$$A_n = 1/4 \left[-(8x^2/n\pi)\cos(n\pi x/8) + 128x/(n^2\pi^2)\sin(n\pi x/8) \right.$$

$$\left. + 1024/(n^3\pi^3)\cos(n\pi x/8) \right]_0^8 - 2 \left[-(8x/n\pi)\cos(n\pi x/8) \right.$$

$$\left. +64/(n^2\pi^2)\sin(n\pi x/8) \right]_0^8 \qquad \text{(C.8-c)}$$

$$A_n = 1/4[-512(-1)^n/(n\pi)+1024(-1)^n/(n^3\pi^3) - 1024/(n^3\pi^3)]$$

$$- 2[-(64/n\pi)(-1)^n] \qquad \text{(C.8-d)}$$

$$An = Bn = 256/(n^3\pi^3)[(-1)^n-1] \qquad \text{(C.8-e)}$$

By substituting A_n and B_n into equation (C.6) the following is obtained.

$$V(x,y) = \frac{256[(-1)^n-1]}{n^3\pi^3} \sum_{n=1}^{\infty} \frac{[\sinh(n\pi y/8)+\sinh\{(n\pi/8)(6-y)\}]}{\sinh(3n\pi/4)}$$

$$\sin(n\pi y/8) \qquad \text{(C.9)}$$

Equation (4.1) is solved by making the variable substitution from equation (C.9) into equation (C.1-a).

$$U(x,y) = (8x - x^2 - 512/\pi^3)$$

$$\sum_{n=1}^{\infty} \frac{\{\sinh[(2n-1)\pi y/8]+\sinh[(2n-1)\pi(6-y)/8]\}}{(2n-1)^3\sinh[3(2n-1)\pi/4]}$$

$$\sin[(2n-1)\pi x/8] \qquad \text{(C.10)}$$

Equation (C.10) was solved by programing the equation into BASIC and running the program on a Z150 PC. It was found that the solution at specific nodal points converged after six iterations to the fifth decimal place. The specific nodal point solutions can be found in Table II.

Appendix D: Solution to the 2-D FFT BVP

The solution to equation (4.1) using the FFT method is simply an extension of the one dimensional problem described in Chapter III. By allowing U(x,y) and F(x,y) to be extended as odd, 2-periodic functions in both x and y, the general form of equation (4.1) is

$$\nabla^2 U(x,y) = -F(x,y) \tag{D.1}$$

and can be represented by the Fourier series

$$F(x,y) = \sum_{j=1}^{M-1} \sum_{k=1}^{N-1} F_{jk} \sin(j\pi x/x_{max}) \sin(k\pi y/y_{max}) \tag{D.2}$$

and

$$U(x,y) = \sum_{j=1}^{M-1} \sum_{k=1}^{N-1} C_{jk} \sin(j\pi x/x_{max}) \sin(k\pi y/y_{max}) \tag{D.3}$$

where C_{jk} is defined as the coefficient of the Fourier series and the mesh size $h_x = x_{max}/(M+1)$ and $h_y = y_{max}/(N+1)$ (17:149). The boundary conditions in equation (D.1) are zero on all boundaries, as are the boundary conditions of equation (4.1). Now, by taking the second derivatives of U(x,y) with respect to x and then y the following is obtained.

$$\frac{\partial^2 U(x,y)}{\partial x^2} = -\sum_{j=1}^{M-1} \sum_{k=1}^{N-1} C_{jk} (j\pi/x_{max})^2 \sin(j\pi x/x_{max}) \sin(k\pi y/y_{max}) \tag{D.4}$$

$$\frac{\partial^2 U(x,y)}{\partial y^2} = -\sum_{j=1}^{M-1} \sum_{k=1}^{N-1} C_{jk} (k\pi/y_{max})^2 \sin(j\pi x/x_{max}) \sin(k\pi y/y_{max}) \tag{D.5}$$

By substituting equations (D.4) and (D.5) into equation (D.1) the following is obtained

$$-\sum_{j=1}^{M-1}\sum_{k=1}^{N-1}\left[(j\pi/x_{max})^2 + (k\pi/y_{max})^2\right] C_{jk}\sin(j\pi x/x_{max})\sin(k\pi y/y_{max})$$

$$= -F(x,y) \qquad (D.6)$$

Solving for C_{jk}

$$C_{jk} = \frac{-F_{jk}}{-[(j\pi/x_{max})^2 + (k\pi/y_{max})^2]} \qquad (D.7)$$

The F_{jk} is computed using the Fourier orthogonality relationship, equation (2.10), and since

$$F(x,y) = \sum_{j=1}^{M-1}\sum_{k=1}^{N-1} F_{jk}\sin(j\pi x/x_{max})\sin(k\pi y/y_{max}) \qquad (D.2)$$

and since, $h_x = x_{max}/(M+1)$ and $h_y = y_{max}/(N+1)$, then

$$F_{jk} = \frac{4(h_x h_y)}{(x_{max} y_{max})}\sum_{m=1}^{M-1}\sum_{n=1}^{N-1} F(x,y)\sin(mj\pi x/x_{max})\sin(nk\pi y/y_{max}) \qquad (D.8)$$

By using equations (D.8), (D.7), and (D.3) the nodal points at any value of U(x,y) can be found and are the three step method discussed in Chapter IV.

Appendix E: Mathematical Explanation of the 2-D Complex FFT

To show that only the imaginary portion of the FFT array must be used for the FFTCC IMSL routine, the following mathematical computation is presented. The FFTCC routine computes the FFT using the form

$$A_{mk} = \sum_{n=0}^{N-1} A_{mn} e^{i2\pi(kn/N)} \qquad (E.1)$$

To compute the two dimensional FFT it is necessary to develop two transforms similar to equation (E.1) using different notation to distinguish between the FFT in the x-direction and the FFT in the y-direction. These two equations are $A^{(1)}$ and $A^{(2)}$ and defined as follows.

$$A^{(1)}_{mk} = \sum_{n=0}^{N2-1} A_{mn} e^{i2\pi(kn/N2)} \qquad (E.2)$$

and

$$A^{(2)}_{jk} = \sum_{m=0}^{N1-1} A_{mk} e^{i2\pi(jm/N1)} \qquad (E.3)$$

where N2 is defined as the number of transformed points in the x-direction and N1 is defined as the number of transforms in the y-direction. $A^{(1)}_{mk}$ is defined as the Fourier Transform of A_{mn} and $A^{(2)}_{jk}$ is defined as the Fourier Transform of $A^{(1)}_{mk}$ and the double Fourier Transform of A_{mn}. To solve for $A^{(2)}_{jk}$, one can substitute equation (E.2) into equation (E.3) and obtain

$$A_{jk}^{(2)} = \sum_{m=0}^{N1-1} \left(\sum_{n=0}^{N2-1} A_{mn} e^{i2\pi(kn/N2)} \right) e^{i2\pi(jm/N1)} \qquad (E.4)$$

Using Euler's identity

$$A_{jk}^{(2)} = \sum_{m=0}^{N1-1} \left\{ \sum_{n=0}^{N2-1} A_{mn}[\cos(2\pi kn/N2) + i\sin(2\pi kn/N2)] \right\}$$

$$[\cos(2\pi jm/N1) + i\sin(2\pi jm/N1)] \qquad (E.5)$$

By multiplying out equation (E.5), the following is obtained.

$$A_{jk}^{(2)} = \sum_{m=0}^{N1-1} \sum_{n=0}^{N2-1} A_{mn}[\cos(2\pi kn/N2)\cos(2\pi jm/N1)$$

$$+ i\sin(2\pi kn/N2)\cos(2\pi jm/N1) + i\sin(2\pi jm/N1)\cos(2\pi kn/N2)$$

$$- \sin(2\pi kn/N2)\sin(2\pi jm/N1)] \qquad (E.6)$$

Note that the only term required to compute the double Fourier sine transform is the last line of equation (E.6), $-\sin(2\pi kn/N2)\sin(2\pi jm/N1)$. It is obvious that the first three cosine and sine terms of equation (E.6) can be eliminated by deleting the real portion of the FFT each time it is computed in the FORTRAN program FFT2D1 (See Appendix B).

Bibliography

1. Brigham, E. Oran. The Fast Fourier Transform. Englewood Cliffs NJ: Prentice-Hall, Inc., 1974.

2. Churchill, Ruel V. and James Ward Brown. Fourier Series and Boundary Value Problems (Third Edition). New York: McGraw-Hill Book Company, 1978.

3. Clark, Melville, Jr. and Kent F. Hansen. Numerical Methods of Reactor Analysis. New York: Academic Press, 1964.

4. Edstrom, Clarence R., Associate Professor. Personal interviews. Air Force Institute of Technology, Wright-Patterson AFB OH, 20 November through 27 November 1985.

5. Fox, Geoffrey C. and Steve W. Otto. "Algorithms for Concurrent Processor," Physics Today, 37 (5): 50-59 (May 1984).

6. Gerald, Curtis F. and Patrick O. Wheatley. Applied Numerical Analysis (Third Edition). Menlo Park CA: Addison-Wesley Publishing Company, 1984.

7. Hamming, Richard W. Introduction to Applied Numerical Analysis. New York: McGraw-Hill Book Company, 1971.

8. Hockney, R.W. "A Fast Direct Solution of Poisson's Equation Using Fourier Analysis," Journal of the Association for Computing Machinery, 12 (1): 95-113 (January 1965).

9. IMSL, Inc. IMSL Library User's Manual (Edition 9.2), Houston TX, November 1984.

10. Rice, Joel. Contractor, Systems and Applied Sciences Corporation. Personal interviews. Air Force Institute of Technology, Wright-Patterson AFB OH, 2 September through 10 December 1985.

11. Roache, Patrick J. "A Pseudo-Spectral FFT Technique for Non-Periodic Problems," Journal of Computational Physics, 27 (2): 204-220 (May 1978).

12. Rosser, Barkley J. Fourier Series in the Computer Age, Contract DA-31-124-ARO(D)-462. Wisconsin University Madison Mathematics Research Center, Madison, Wisconsin, February 1974 (AD-775 585/3).

13. Skollermo, Gunilla. "A Fourier Method for the Numerical Solution of Poisson's Equation," Mathematics of Computation, 29 (131): 697-711 (July 1975).

14. Smith, Gordon D. Numerical Solution of Partial Differential Equations. London: Oxford University Press, 1965.

15. Titchmarch, Edward Charles. Introduction to the Theory of Fourier Integrals. Oxford: The Clarendon Press, 1948.

16. Vichnevetsky, Robert. Computer Methods for Partial Differential Equations, Volume 1. Englewood Cliffs NJ: Prentice-Hall, Inc., 1981.

17. Weinberger, Hans F. A First Course in Partial Differential Equations with Complex Variables and Transform Methods. New York: Blaisdell Publishing Company, 1965.

Vita

Major Todd R. Jones was born on 14 September 1951 in Malad, Idaho. He attended the United States Military Academy, West Point, New York from which he received the degree of Bachelor of Science and was commissioned a Second Lieutenant in the U.S. Army in 1973. Upon graduation, he served in the Field Artillery until being accepted for the Army's Flight Training Program in 1975. Since graduation from Flight School, Major Jones has served in several leadership positions, both in Aviation and in Field Artillery. He served as the S-3 of the 8th Combat Aviation Battalion, Mainz, West Germany, prior to entering the School of Engineering, Air Force Institute of Technology, in August 1984.

Permanent address: 2629 Meadow Avenue
Caldwell, Idaho 83605

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

AD-A173 917

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release distribution unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) AFIT/GNE/ENP/86M-8	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION School of Engineering 6b. OFFICE SYMBOL (If applicable) AFIT/ENP	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State and ZIP Code) Air Force Institute of Technology Wright Patterson AFB, Ohio 45433	7b. ADDRESS (City, State and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State and ZIP Code)	10. SOURCE OF FUNDING NOS. PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT NO.
11. TITLE (Include Security Classification) See Box 19	
12. PERSONAL AUTHOR(S) Todd R. Jones, B.S., MAJ, US ARMY	
13a. TYPE OF REPORT MS Thesis 13b. TIME COVERED FROM _____ TO _____	14. DATE OF REPORT (Yr., Mo., Day) 1986 March 15. PAGE COUNT 79
16. SUPPLEMENTARY NOTATION	
17. COSATI CODES FIELD / GROUP / SUB. GR.	18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number) Fast Fourier Transforms, Boundary Value Problems Poisson's Equation, Finite Difference Method

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Title : FEASIBILITY AND COMPARISON INVESTIGATION OF THE USE OF THE FAST FOURIER TRANSFORM AND FINITE DIFFERENCE METHOD FOR NUMERICAL SOLUTION OF BOUNDARY VALUE PROBLEMS

Thesis Chairman: Dr. Bernard Kaplan
Professor of Physics

Approved for public release: IAW AFR 190-1.
LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology (AFIT)
Wright-Patterson AFB OH 45433
9 May 86

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL Dr Bernard Kaplan	22b. TELEPHONE NUMBER (Include Area Code) 513-255-4498 22c. OFFICE SYMBOL AFIT/ENP

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

The purpose of this study was to determine the feasibility of using Fast Fourier Transforms (FFT) to solve Boundary Value Problems (BVP) and then compare the results to those of the Finite Difference Method (FDM). Variations of Poisson's one and two dimensional equations were used as a vehicle to develop the FFT method. For the one dimensional BVP, both homogeneous and non-homogeneous Dirichlet boundary conditions were considered. In the one dimensional BVP the inhomogeneous function, F(x), was also varied. The two dimensional BVP, only one inhomogeneous function, F(x,y), and homogeneous boundary conditions were used. The one dimensional model was used as a basis for developing the two dimensional model.

The analytical solution of each problem was compared to the numerical solution of the FDM and the FFT method at varying mesh sizes. The computational time of the FDM and the FFT method were also compared.

The results indicate that the FFT is extremely efficient in the two dimensional BVP because of the computer storage space required and the computational time needed to solve the FFTs. The accuracy of the FFT compares favorably to the FDM and, as the mesh size decreases, becomes more accurate than the FDM.

END

11-86

DTIC